AD-A104 140 ROYAL AIRCRAFT ESTABLISHMENT FARNBOROUGH (ENGLAND) F/G 9/2
PRODUCTION OF CHEMICAL STRUCTURE DRAWINGS USING AN INTERACTIVE --ETC(U)
FEB 81 G SHARPLES
UNCLASSIFIED RAE-TR-81030 DRIC-BR-79132 NL

1 OF 1
AD
A104140

END
DATE
FILMED
10-81
DTIC

TR 81030

ROYAL AIRCRAFT ESTABLISHMENT

*

Technical Report 81030

February 1981

PRODUCTION OF CHEMICAL STRUCTURE DRAWINGS USING AN INTERACTIVE GRAPHICS SYSTEM

by

G. Sharples

*

Procurement Executive, Ministry of Defence
Farnborough, Hants

(14) RAE-TR-81Ø-1

UDC 541.6 : 681.3 : 518.4 : 519.688

(1) DRIC (19) BR-7112-

ROYAL AIRCRAFT ESTABLISHMENT

(1) Technical Report 81030

Received for printing 25 February 1981

(6)

PRODUCTION OF CHEMICAL STRUCTURE DRAWINGS USING AN INTERACTIVE GRAPHICS SYSTEM

by

(10) G./Sharples

(11) Feb 81 (12) 57

SUMMARY

This Report describes a series of computer programs which allows drawings of complex chemical structures to be displayed on a graphics terminal, to be modified using the graphical input devices and finally, to be drawn by an incremental plotter. The system thus provides a means for the interactive development of chemical structure diagrams and for the production of high quality drawings suitable for inclusion in published reports.

The system is based on the graphical definition of several hundred chemical groups. The structure of more complex compounds can be built up from these basic units and displayed to the user. Optional features of the system include variation of the scale of drawing, interactive modification of the drawings using a light pen and automatic detection and prevention of drawing overlaps.

Departmental Reference: IT 171

LIST OF CONTENTS

		Page
1	INTRODUCTION	3
2	AN OVERVIEW OF THE STRUCTURE DISPLAY PROGRAM	3
3	THE HARDWARE AND SOFTWARE ENVIRONMENT	5
4	THE SHAPE DEFINITION FILE	5
5	THE SHAPE DEFINITION PROGRAM	7
6	THE STRUCTURE DEFINITION FILE	7
7	THE METHOD OF SHAPE CONNECTION	8
8	THE STRUCTURE DISPLAY PROGRAM	10
	8.1 Initialisation phase	10
	8.2 Setting the program environment	10
	8.3 Shape processing	14
9	THE HARD COPY PROGRAM	15
10	PROGRAM LIMITATIONS	17
11	CONCLUSION	18
Acknowledgments		18
Appendix A	Format of the shape definition data	19
Appendix B	Format of the structure definition data	25
Appendix C	Features of the graphics software	28
Appendix D	A description of the structure display program in execution	29
Appendix E	Structure display program flowcharts	32
Appendix F	Execution exception conditions	39
Table 1	Order of search for a match between the bonds of a link shape and the current shape	41
References		42
Illustrations		Figures 1-13
Report documentation page		inside back cover

1 INTRODUCTION

This Report describes a sequence of computer programs which allows drawings of complex chemical structures to be displayed on a graphics terminal, to be modified using the graphical input devices and finally to be drawn by an incremental plotter. The system thus provides a means for the interactive development of chemical structure diagrams and for the production of high quality drawings suitable for inclusion in published reports.

The programs were developed in response to a request from Materials Department at RAE, Farnborough. The Polymer Chemistry Section has developed a software system which examines the properties of a set of polymers and calculates overall property coefficients for the constituent chemical groups. These coefficients can then be used to predict the bulk properties of chemical combinations of the groups and, in theory, to predict which combinations of groups would meet a particular requirement. The most important polymer property in this context is the glass transition temperature which is the temperature at which a polymer changes from the glassy to the rubbery state.

The analysis programs operate on a collection of approximately 350 groups and these form the basic building blocks of the system. Each group is allocated a unique group number. A higher level group can be constructed from three or more basic groups and this group also has an identity number. These identity numbers are not yet universally standardised and so it is necessary to publish not only the results of the calculations, but also a chemical representation of the higher level groups which is readily intelligible and universally understood, *ie* a chemical drawing. Since the results for many thousands of groups and polymers need to be published, the work involved in the production of the drawings by hand would be prodigious. This Report describes a means of automating the drawing process.

Examples of drawings produced by the programs are shown in Figs 7 to 13.

2 AN OVERVIEW OF THE STRUCTURE DISPLAY PROGRAM

Within this Report, a basic group is referred to as a *shape*, since it is the basic unit of graphical manipulation, and the more complex groups and polymers are referred to as *structures*.

The structure display program was designed to operate using the same input data as is presented to the analysis programs. The data definition of a structure consists of a series of shape numbers structured so that the data not only defines the constituents of the structure, but also how the constituents are linked together to form *chains* and how chains are connected together to form more complex chemical structures. Since the input data representation was fixed, the tasks involved in system development were:

(1) to define a data convention which could be used to represent each of the shapes in graphical terms (see section 4);

(2) to represent all of the shapes in this data format;

(3) to define the topological rules by which shapes are connected into chains and by which chains are connected together to form structures (see section 7);

(4) To write a program to interpret the input data and the graphical data, to implement the connection rules and to produce the chemical drawings.

Each shape is defined graphically as a combination of lines, strings of full sized characters and strings of half sized characters (see Fig 1). Shapes are connected together by a straight line known as a *bond* and one shape can have several bonds. Each shape definition contains, for each bond, the coordinates of the end of the bond nearest its parent shape together with the angle between the bond and the horizontal. Bonds are normally of fixed length. Since a bond in one shape must connect in a straight line to a bond in the next shape, the program must be able to shift and rotate the shapes in order to match the bonds in position and in angle (see Fig 2).

The task of constructing the data file containing the graphical definition of all the shapes was undertaken jointly by RAE and by The Rubber and Plastics Research Association. This file is known as the shape definition file.

There are three programs in the complete chemical drawing system. The shape definition program preprocesses the shape definition data into a readily accessible form (see section 5); the structure display program connects shapes together and displays the structures on an interactive graphics terminal; and the hard copy program converts the output from the structure display program into a form suitable for the production of high quality drawings on an incremental plotter (see section 9).

The structure display program reads the data definition of a structure, analyses it to extract the shape numbers which make up the structure and accesses the shape definition data to obtain the graphical representation of each shape. Each representation is displayed suitably positioned and rotated so that one bond connects in a straight line to a bond of the previous shape. Given that a shape can have bonds pointing in several directions and the next shape to be connected to it can also have several bonds, the rules by which shapes are connected together are necessarily rather complex in order to cope with all possible situations. In general, the structures are displayed as orthogonal chains, and bonds with intermediate angles are only used when no horizontal or vertical bond is available.

The program offers the user a choice of several options in order to match what the program does to the user's current requirements. The options offered are:

(a) Interactive mode. The user can be given the opportunity to change the bond lengths of any shape in the structure displayed, in order to improve its appearance or to prevent overlaps between adjacent shapes.

(b) Copy mode. Once a drawing has been accepted by the user, a definition of the drawing can be sent to a disk file and this can later be processed to produce a permanent and high quality copy on an incremental plotter.

(c) An alternative algorithm for shape connection. A slightly different connection algorithm can be selected if the algorithm normally used by the program is not producing satisfactory results for individual structures. This alternative algorithm uses less 'built-in logic' and gives more weighting to the way the user has supplied the structure definitions.

(d) Selective mode. Individual structures can be picked out for display rather than processing all the structures in the structure definition file.

(e) Overlap mode. The program can check whether drawing overlaps occur between adjacent shapes and if so, it will modify the connecting bond in an attempt to prevent the overlap.

(f) Graphics mode. The program can display on a graphics terminal all of the structures as they are processed. Up to nine structures can be displayed at once.

(g) Scale. The scale of drawing can be modified to magnify small structures or to reduce the size of the large structures.

These options are presented to the user in the form of a menu displayed on the graphics terminal and they can be made active or dormant at any time by use of a light pen in association with the menu.

The program thus provides the user with a powerful array of facilities to develop visually attractive chemical drawings and the man/computer interface is designed to be easy to use for those chemists who have little computer experience.

Appendix A contains an example of shape definition data, Appendix B contains examples of structure definition data and Figs 7 to 13 give examples of drawings produced on the incremental plotter.

3 THE HARDWARE AND SOFTWARE ENVIRONMENT

The programs run on a PDP11 minicomputer under the RSX11M operating system[1] and they are written in FORTRAN. The structure display program occupies slightly less than 32K words of memory.

The graphics terminal is a Vector General, refreshed 'line' display, with 16K of memory in the interface from which the picture on the screen is refreshed. The program makes use of the keyboard associated with the display to take in alphanumeric information and of the light pen to allow the user to identify lines and menu items on the screen.

The graphics software used is the General Purpose Graphics System (GPGS)[2,3]. The package consists of a number of device drivers together with a device independent library of FORTRAN callable routines. More details of the features of GPGS which are used by the programs are given in Appendix C.

4 THE SHAPE DEFINITION FILE

Each chemical shape is defined by a shape number and four categories of information:

(1) the lines making up the shape;

(2) the strings of full sized characters annotating the shape;

(3) the strings of half sized characters providing further annotation;

(4) the positions of the connecting bonds and their angles to the horizontal.

This information is specified as a series of fixed format records and the records must occur in the order:

line data
full sized text data
half sized text data
bond data.

Since many chemical shapes are similar in graphical structure a category of data need only be supplied if the data for the current shape differs from that for the same category of the previous shape, *ie* if a category of data is omitted, the corresponding data for the previous shape is assumed to be repeated. The shape definitions can be supplied in any order, *ie* not necessarily in shape number order, and so similar shapes can be grouped together to simplify this task of shape definition. It is important to note that omitting a category of data does not mean that no data is defined for this category. If no data is to be defined, an explicit 'no data' record specifying zero shape lines or character strings must be provided, unless the previous shape had a 'no data' record for this category.

The line data is provided as a series of relative (X,Y) coordinates together with indications of whether the resulting lines are to be visible or invisible; the character strings are specified as collections of characters, one string per record; and the bond data for each bond consists of the absolute (X,Y) coordinates of the end point nearest the parent shape together with the angle of the bond to the horizontal. Valid bond angles range from 0^o to 345^o in anticlockwise increments of 15^o. A detailed specification of the shape data format is given in Appendix A, together with some examples.

When a shape is rotated, the centre point of each character string is rotated to shift the position of the string but the text remains horizontal, *ie* relative to the lines making up the shape, the text is rotated about the centre point of the string. The resulting relationship between the text and the lines might not be visually satisfactory and interference can occur between the lines and the text (see Fig 4b). The program provides two facilities to aid the user in avoiding some of the worst instances which occur during shape rotation.

The user can define a *rotation relation* for the current shape. This is a shape which, when displayed, looks like the current shape rotated through 90^o and thus the program can use the data representation of the rotation relation instead of performing a strict $+90^o$ or -90^o rotation on the data representation of the current shape. For example, shapes 6 and 7 are rotation relations (see Fig 4a).

The rotation relation can either be a shape which is defined elsewhere in the shape definition file and has a shape number, or the definition can immediately follow the definition of the current shape, in which case it has no separate shape number. The use cf a rotation relation guarantees that the shape drawing which results from a $+90^o$ or -90^o rotation will be aesthetically acceptable.

Secondly, the user can direct the program to maintain the relative positions of two or more character strings during rotation. This is always necessary when the text is defined to create subscripts, *eg* if the shape is to be annotated with the text 'CH_2', this must be defined as two strings, 'CH' and '2', and the start position of the string '2' must be fixed relative to the start position of the string 'CH' if the rotated versions are to be visually acceptable. The user can request in a string definition that the current string is to be 'attached' to the preceding string for the same shape and then the relative positions of the two shapes are always maintained.

The order in which the bonds are specified can affect the layout of a drawing. When attempting to match the bonds of two shapes, the bonds are examined in the order in which they are specified in the shape definition data and so the bond which is defined first is given preference. Furthermore, whenever the structure display program has to make an arbitrary selection of a bond, it selects the next available bond, again taking the bonds in the order in which they were originally defined. In general, bonds should be defined, ordered according to the preferred directions of development. Section 7 gives further details on the method of shape connection.

5 THE SHAPE DEFINITION PROGRAM

Running on a PDP11 minicomputer, there is insufficient program address space to retain all the shape definitions in immediately accessible form in memory and the shape definitions are thus stored in a random access file on disk. A preprocessor program, the shape definition program, reads the user's shape definition file and produces the random access file to be used by the drawing program. The drawing program can easily bring shape definitions into memory when they are required but an overhead in accessing the disk unit is involved.

If the format conventions of the shape definition data are broken, the program prints an error message indicating the contravention which it has detected. Appendix F contains a list of the error messages which can be produced during execution of the shape definition program.

6 THE STRUCTURE DEFINITION FILE

The structure display program produces drawings in accordance with instructions read from the structure definition file. Each structure in this file is defined by a text descriptor record and by one or more chain records. The text descriptor record simply contains alphanumeric text describing the structure to be drawn, *eg* its identification and perhaps its major chemical attribute. Each chain record consists of a series of shape numbers separated by hyphens indicating which shapes make up the chain. An example of such a chain record is

-49-6-163

This data defines a chain consisting of shape 49, shape 6 attached to shape 49 and shape 163 attached to shape 6. Since each structure can consist of several linked chains an asterisk is included after the shape number whenever a shape is to form a junction between two chains. Such a junction shape is known as a *link shape* and the number of asterisks indicates how many chains are to be connected to that link shape. For example,

-19**-184**-6

indicates that two chains are to be connected to shape 19 and two chains are to be connected to shape 184. Subsequent records define the *side chains* which are to be attached to these shapes and these records must be ordered to correspond with the order in which the asterisks are encountered, reading records sequentially and processing characters from left to right. The order of side chain records for the above would be:

first side chain to be attached to shape 19
second side chain to be attached to shape 19
first side chain to be attached to shape 184
second side chain to be attached to shape 184.

As an extra confirmation of the order of the records the first shape number in each side chain record is the shape number of the link shape to which it is to be connected and it is not a request for a shape to be drawn. Thus a partial structure definition might be:

```
GROUP 9999 HAS 4 SIDE CHAINS
-19**-184**-6
-19-1
-19-174
-184-36
-184-49-103*-49
```

Here, one of the side chains has a further side chain attached to it.

The program maintains a 'first in - first out' stack of all link shape requests and so very complex structures can be drawn.

If a side chain record contains only the number of the associated link shape, the side chain is called a *dummy side chain*. Dummy side chains can be used to force the program to abandon its preferred method of linking chains together and to adopt a sequence preferred by the user. A dummy side chain effectively links a null shape to the first available bond of the link shape concerned. The null shape cannot be seen but the connecting bond is marked as 'occupied' by the program and is not available for subsequent linking. Fig 3 contains examples of how the use of link shapes and dummy side chains can be used to modify the appearance of a drawing.

Appendix B contains further details on the format of structure definition data together with some examples.

7 THE METHOD OF SHAPE CONNECTION

The data contained in the structure definition file does not imply a unique method of drawing each structure defined in it. The program still has many options as to how each shape can be connected to its neighbour and how chains of shapes can be connected to the link shapes.

In the following description, the bond of the current shape which is attached to the previous shape is referred to as the *entry bond* of the current shape and the bond to which it is connected is known as the *exit bond* of the previous shape.

The program performs the following procedure in order to construct a chain of linked shapes:

(1) It draws the first shape, perhaps rotated to create a horizontal bond, and attempts to add succeeding shapes in a horizontal, rightward direction or in a vertical, downward direction.

(2) It adds the next shape in the current direction, *ie* it selects an exit bond opposite to (180° out of phase with) the entry bond. If no such bond exists, it selects the first unused bond as the exit bond.

(3) It selects the entry bond of the current shape by considering the rotation necessary to connect each potential entry bond to the selected exit bond. The program contains a list of bond phase differences, stored in order of preference. A 180° phase difference is the first preference since no rotation would be required to connect the two shapes; a 0° phase difference is the second preference, in which case a mirror image rotation would be required; and the next two preferences are 90° and 270° when a right angled rotation would be required. These are followed by the other possible phase differences in a rather arbitrary order:

210° 30° 120° 300° 225° 45° 135° 315° 240° 60° 150° 330°

Using this hierarchy of phase differences, each bond of the current shape is examined first of all for a 180° phase difference with the selected exit bond. If no such match is found the program looks for a 0° phase difference, and so on down the hierarchy until a match is found and the entry bond has thus been selected.

This procedure defines how chains are constructed but a further mechanism is required when a new chain is started, attached to a previously drawn link shape. In this case, the program has even more options open since there is no immediately obvious way of identifying which potential exit bond from the link shape should be used. If the 'take link shape exits sequentially' option is enabled (see section 8.2) the program merely takes the next free bond of the link shape as the exit bond and the matching procedure is then identical to (3) above. If the option is disabled, the program

(a) takes the first free bond of the link shape;

(b) examines each bond of the current shape for a 180° phase difference;

(c) if no match is found, it examines each bond of the current shape for 0°, 90° and 270° phase differences in succession, until a match is found;

There is thus a strong bias in favour of using the first free bond of the link shape as the exit bond and this bond will be selected when the connection can be made using a rotation of a multiple of 90°.

(d) if still no match is found, the program repeats (b) and (c) for successive bonds of the link shape until a match is found;

(e) if no match is found, the program repeats (a), (b), (c) and (d) but using phase differences 210°, 30°, 120° and 300° successively;

(f) if no match is found, the program repeats (a), (b), (c) and (d) but using phase differences 225°, 45°, 135° and 315° successively;

(g) if no match is found, the program repeats (a), (b), (c) and (d) but using phase differences 240°, 60°, 150° and 330° successively.

Again, the search priority is arbitrary after the first four phase differences. The search priority can be represented in tabular form and Table 1 gives the order of priority for a link shape with three free bonds to be connected to the current shape which has two bonds.

8 THE STRUCTURE DISPLAY PROGRAM

8.1 Initialisation phase

The structure display program poses a series of initial questions in order to obtain the basic information necessary to start the program run:

(a) DATA FILE NAME?

The user replies by typing on the terminal the name of the file containing the definition of the structures to be drawn,
eg

```
STRUCTS.DAT<RETURN>
```

or

```
DK2:[1,6]POLYMS.DAT<RETURN>
```

(b) COPY FILE NAME?

The program requires the name of the output file which is to contain the drawing definitions to be used for the production of hard copies. The file name is only used if the hard copy option is activated and so the response is irrelevant if the user does not intend to produce hard copy output. However, the reply must still be a valid RSX11M file name or an error message is produced by the RSX11M file system.

If a file of the given name already exists, a new generation of the file is created and the old version is untouched. Examples of valid responses are

```
X<RETURN>
STRUCTS.COP<RETURN>
```

(c) SCREEN ARRAY SIZE?

The user can choose to collect the drawings into 1, 4 or 9 element arrays. The reply to this question can thus be

```
1<RETURN>
2<RETURN>
```

or

```
3<RETURN>
```

to indicate a 1 × 1, 2 × 2 or 3 × 3 matrix of drawings. Any other value will be rejected and the question will be asked again. When using the Vector General display, the larger the array size, the smaller will be the size of the individual drawings displayed on the screen.

8.2 Setting the program environment

The program reads the first array of structures from the structure definition file and displays the drawings on the Vector General screen. In addition it displays a menu of environment options across the top of the screen. An option can be switched on and off alternately by 'picking' the corresponding text with the light pen. An option which is activated has a '#' sign displayed by the side of the corresponding menu item.

A menu item is picked by pointing the light pen at the appropriate text. When the pen can 'see' the text, a 'Δ' symbol is displayed beneath the text and the selection can then be confirmed by touching the metal tip of the pen with a finger of the hand holding the pen.

The options available are:

(a) Copy mode (menu item COPY): when the option is enabled, the program produces an output file containing a definition of all drawings processed. This output file can later be processed to produce hard copy output (see section 9). This facility is used for production runs.

(b) Trace (menu item TRACE): when this option is selected the program prints out a short piece of text together with a series of values whenever a significant program decision is taken. This facility is not intended for general use but is an aid to be used when debugging the program.

(c) Batch mode (menu item BATCH): the program can either operate in interactive mode, where the user has the opportunity to examine each drawing and to make modifications to it; or in batch mode, where the program does not pause between drawings, but processes a whole series, one after the other. If the user activates the BATCH option, the question is asked:

```
BATCH SIZE (I3) ?
```

to which the user replies, on the Vector General keyboard, with three characters (spaces and/or right justified numbers) specifying the number of arrays of structures to be processed in the batch. For example, valid replies are:

```
123 ⟨RETURN⟩
∇12 ⟨RETURN⟩
∇∇9 ⟨RETURN⟩
```

If a negative number is returned, then the program processes all the drawings in the structure definition file.

(d) Graphics on Vector General display (menu item VG): when the drawings are to be processed in BATCH mode, it is often convenient to suppress the display of the structures on the Vector General screen. Selection of this menu item causes all drawing instructions to be ignored, the menu is removed, preventing all further interaction, and all of the remainder of the structure definition file is processed as a batch.

(e) Link shape exits to be used sequentially (menu item SEQ EXIT): the program offers two methods of connecting the next shape to be processed onto a previously drawn link shape (see section 7). If this option is selected then the bonds for the link shape are used sequentially, *ie* in the order in which they were specified in the shape definition file, and the most suitable bond of the next shape is fitted onto the next free bond of the link shape. If the default algorithm is used, then each of the link shape bonds is compared with each of the bonds of the next shape, and the most suitable pairing is selected. These two alternatives allow the program to use its intelligence in drawing the majority of the structures, but for a minority of awkward cases any required drawing scheme can be imposed to over-ride the preference built into the program.

(f) Overlap check (menu item OVERLAP): the program offers the option of automatic detection and attempted correction of any overlaps which occur when drawing the structures. If this option is selected the program retains the maximum and minimum X and Y coordinates of each shape drawn and before drawing the current shape, it checks whether there is any overlap between the rectangle containing the current shape and the rectangles containing the earlier shapes. If there is an overlap, it attempts to avoid the difficulty by first shortening the entry bond, and then incrementing its length up to 14 times the standard bond length. If all overlap is prevented the shape is drawn with the extended bond; if overlap still occurs it is drawn with a bond length 15 times the standard bond length and it is left to the user to remove the overlap during the interactive phase. The overlap check never operates on a bond whose length has been specified by the user during the interaction phase and so the program cannot over-rule the user's requests. This interference check is rather crude, and the corrective logic keeps shapes further apart than is absolutely necessary. In addition, the remedy which it applies only acts on the entry bond to the current shape and not on an earlier shape which might be the real cause of the problem. However, it is effective for the majority of simple cases of overlap.

(g) Selective mode (menu item SELECT): this option allows the user to select individual structures, one at a time, from the data file or to select a starting point within the data file. If this option is activated, then the program asks

```
NEXT GROUP ? - 4 CHARS
```

The program expects the user to type four characters which define the next (or first) structure to be processed. The four characters received are compared with all four character strings in the text descriptors associated with successive records from the structure definition file and processing only starts when a match has been found somewhere within a text descriptor. The four character identifier can generally be supplied in 'free format' if the significant items in the text descriptors which are used to identify structures are followed by space characters. If the user supplies less than four characters, then the number is made up to four by the addition of trailing spaces.

Thus,

10⟨RETURN⟩ is equivalent to 10∇∇⟨RETURN⟩

and this will be matched with a string '10∇∇' in a text descriptor. Similarly, a reply of ⟨RETURN⟩ alone is equivalent to a reply of 4 spaces. This will be matched with the next text descriptor (unless the text descriptor contains more than 31 characters without four consecutive spaces) and the next structure is processed. In the RAE application, the text descriptors contain the group number, right justified. Thus a reply of

∇890⟨RETURN⟩

indicates that group number 890 is the next group to be processed and the structure definition file is searched sequentially until this group is found.

If BATCH mode has been selected, selective mode is turned off automatically and processing continues from this point. The record indicated is thus the first record of the batch. If BATCH mode has not been selected, the program returns after the structure has been accepted by the user to ask:

NEXT GROUP ? - 4 CHARS

and the cycle is repeated. The user can select another structure which is defined further down the data file. Thus structures can be processed selectively, one at a time and in the order in which they are defined in the data file. If the option is switched off on any occasion, the program returns to sequential, interactive processing and the record selected is thus the starting point for an interactive session.

If the program fails to find the required text descriptor, the program terminates when all of the structure definition file has been read.

(h) Change scale (menu item SCALE): the initial drawing area for a structure is 300 units by 300 units. For the display of a single shape, this area is often too large and the drawing would benefit from scaling up; if for the display of a complex polymer, the area is too small, data is lost over the edges and the drawing would benefit from scaling down. When the SCALE menu item is selected the program displays the message:

TYPE SCALE FACTOR (F 3.1)

and the user replies with three characters, including a decimal point, on the Vector General keyboard to indicate the desired scaling factor, *eg*

2.0 ⟨RETURN⟩

or

0.5 ⟨RETURN⟩

The initial drawing area is divided by the factor supplied to give the current drawing area. Thus a factor of 2.0 indicates that the drawing area mapped onto the screen is halved in the X and Y directions and the size of the drawing is doubled. The new scale remains in force until the SCALE menu item is selected again. Returning a factor of 1.0 restores the drawing area to its default state.

(i) Redraw the current array (menu item REDRAW): until a structure has been drawn the program cannot know the bounds of the drawing. Having once drawn it, the program knows how far to offset the drawing of each structure in order to centre it in the drawing area and the REDRAW menu item allows the user to request that the current array of structures be re-displayed with each structure centred in its drawing area.

(j) Cease execution (menu item STOP): Program execution can be terminated by selecting the STOP menu item.

Menu items can be selected at any time but the selection is only activated when the drawing of the current array of structures has been completed. Selection of the

TRACE, SEQ EXITS, OVERLAP or SCALE

menu items causes the current array of structures to be re-drawn with the new option in force.

In the initial program environment, the overlap check is activated, and copy mode and selective mode are switched off.

8.3 Shape processing

The structure display program reads the structure definitions from the input data file and sends the drawing instructions to the Vector General display and/or to a 'copy' output file. As each structure definition is read, the program prints the associated text descriptor on the terminal as a permanent record of which structures have been processed.

In BATCH mode no action is required from the user. The program continues processing structures until the required batch size has been processed or until the structure definition file has been exhausted. In both cases, the program ceases execution and is removed from memory.

In interactive mode, the user can examine each array of structures after it has been displayed, and then optionally modify any of the bond lengths. When a user response is required the program rings the terminal bell as a prompt and the user then has the option:

(a) to accept the array of structures as displayed and to proceed to the next structure. In this case, the user just replies with:

⟨RETURN⟩

on the Vector General keyboard, and the program rings the terminal bell again to indicate that the interaction has been accepted;

(b) to change the length of a bond in any of the structures displayed for the purposes of improving the appearance of the structure. The user points the light pen at the bond in question. When the pen is pointing directly at a bond, a 'Δ' symbol is displayed at the end of the bond nearest to its parent shape. To trigger the interaction, the user touches the metal tip of the pen with a finger of the hand holding the pen whilst the pen is pointing at the required bond. The program rings the terminal bell to indicate that the interaction has been accepted and the user then types a single digit number (0 to 9) and ⟨RETURN⟩ on the Vector General keyboard. The program multiplies this number by the standard bond length to determine the requested bond length whenever the selected bond is drawn. Thus the separation between shapes can be varied at the user's request between zero and 9 times the standard separation, and all changes in bond length are faithfully reproduced in the 'copy' output file. Alterations in bond lengths can be used to avoid shape overlaps when complex shapes are being drawn.

After the amended picture has been drawn, the user is again faced by the two options (a) and (b) described above and this interactive loop is pursued until option (a) is selected.

Appendix D gives a step by step description of the structure display program as it displays a chemical structure and Appendix E contains the program flowcharts. Appendix F contains a list of error messages which can be produced during execution.

9 THE HARD COPY PROGRAM

The hard copy program reads the drawing definition file created by the structure display program and produces a magnetic tape containing drawing instructions in a format suitable for processing on a Calcomp 905/936 plotting system. The plotter then produces drawings identical to those displayed on the Vector General screen.

The program asks for the name of the drawing definition file:

```
COPY FILE NAME ?
```

and the user replies with the name of the file which was supplied to the structure display program for use as the 'copy' file.

If the magnetic tape unit is not 'ONLINE' with a magnetic tape positioned at the beginning of tape marker and write enabled, the program produces the message:

```
MAG TAPE NOT READY - PRESS RETURN TO RETRY
```

When the user replies with

⟨RETURN⟩

on the terminal, the program checks the status of the magnetic tape unit and then either repeats the message or initialises the magnetic tape to receive the drawing instructions.

The program instructs the user:

TYPE IN PLOT SIZE IN CMS (F3.0)

The reply consists of two figures separated by a decimal point, *eg*

4.5 ⟨RETURN⟩

or

8.0 ⟨RETURN⟩

The number provided by the user defines a square which is to contain the plot of one array of structures as displayed on the Vector General screen. The program further needs to establish how the collection of these square plots is to be laid out on the paper. The squares can be collected together to form a 'page' and the pages can be collected in columns so that a minimum of plotting paper is used. The program instructs:

TYPE NO. ACROSS & NO. DOWN/PAGE(2I2)

and it expects a reply of 2 two digit numbers or 2 one digit numbers with leading spaces to specify how the drawings are collected into pages. Examples of valid replies are:

1010⟨RETURN⟩	*ie*	10 across and 10 down
∇2∇3⟨RETURN⟩	*ie*	2 across and 3 down

The next instruction is:

TYPE NO. OF PAGES DOWN (I2)

and the program expects a reply of a two digit number or one digit number with a leading space to specify how many pages are to be drawn in a column, *eg* valid replies are:

10⟨RETURN⟩	*ie*	10 pages/column
∇3⟨RETURN⟩	*ie*	3 pages/column

The drawing area of each plot is delimited by the lines of the corresponding square, each plot within a page is separated from its neighbours by a gap of 0.5 cm and pages are separated by a gap of 4 cm. Since the width of the plotter paper is 86 cm, the user should choose values such that

$$p(sd + 0.5(d - 1)) + 4(p - 1) \leqslant 86$$

where d is the number of plots vertically/page
p is the number of pages/column
s is the size of the plot in centimetres.

The values used to produce the drawings in Appendix C are:

the plot size is 8.0 cm
a page consists of two plots across and three down
three pages are plotted per column.

At the start of each plot the program rings the terminal bell to indicate to the user how quickly the drawings are being processed. In addition, at every tenth plot, it prints the total number of plots processed so far so that the user has an approximate record of how many plots are stored on the magnetic tape.

The contents and validity of a drawing definition file can quickly be checked by running a different program which reproduces the drawings on the Vector General screen.

10 PROGRAM LIMITATIONS

The size of various data fields are fixed within the program, mainly by declaring fixed size arrays in the FORTRAN code and this imposes limitations on the scope of problem which can be handled. The main limitations are:

(1) Each shape definition can have up to

(a) 83 X and Y coordinate pairs

(b) 15 full sized text strings of up to 5 characters

(c) 2 half sized text strings of up to 15 characters

(d) 10 bonds with angles 0^o to 360^o in 15^o increments.

(2) The maximum number of shapes which can be defined is 400.

(3) The maximum shape number is 500.

(4) Up to 20 link shapes can be stored on the 'future link shape' stack.

(5) Up to 40 bonds can have a non-standard length defined by the user in any array of structures.

(6) Each structure is displayed by default in a 300 × 300 display area. Data appearing outside this area is normally not displayed and lines are clipped at the boundary.

(7) An array of structures can be defined by up to 100 structure description records consisting of up to 1500 non-space characters.

(8) Eight shape descriptions are maintained in the cyclic buffer.

(9) When the overlap option is used, a structure may consist of up to 100 shapes.

(10) A text descriptor record can consist of up to 35 characters.

In addition, certain fixed values are written into the program:

(1) The standard bond length is 5 units and thus shapes are normally separated by 10 units.

(2) Full sized characters are drawn on a 6 × 7 matrix but commonly only the first five units in X are used to allow for the spacing between characters.

(3) Half sized characters are drawn 4/7 the size of full sized characters.

11 CONCLUSION

The objective in writing this system was to take the data already in use with the polymer analysis programs and to produce visually attractive chemical diagrams representing the structures defined. Much initial effort has been expended in the definition of the shape data and ın the development of the program to display the diagrams. The resulting programs and data now constitute a production tool which can be used to create chemical structure drawings in large numbers. Furthermore, the system is flexible in that shape definitions are easily added to the shape definition file and so the application of the system is much wider than the area of polymer chemistry.

Acknowledgments

Grateful thanks are due to Dr W.A. Lee of Materials Department who has provided the expertise in chemistry which was necessary to produce the system definition and has supported the project with consistent enthusiasm throughout its development.

The assistance of The Rubber and Plastics Research Association of Great Britain in the preparation of the shape definition data is also gratefully acknowledged.

Appendix A

FORMAT OF THE SHAPE DEFINITION DATA

A.1 The format specification

The data for each shape defines:

(a) the shape number

(b) the lines making up the shape

(c) strings of full sized characters and their positions

(d) strings of half sized characters and their positions

(e) the positions of the connecting bonds and their angles to the horizontal.

The shape definitions can be supplied in any order and a category of data need only be supplied if the data for the current shape differs from the same category for the previous shape.

The first record for a shape contains a space character (column 1) and the shape number (columns 2 to 4).

The line data is provided as a record containing an 'S' (column 1) and the number of lines in the shape (columns 6 to 8), followed by a series of records giving the X-Y coordinates. Each X-Y coordinate pair defines a line relative to the coordinates of the endpoint of the previous line, or relative to the point (0,0) for the first line of a shape. Each record contains up to seven pairs of relative coordinates and the format is:

```
Columns: (12 to 15, 16 to 19)  (22 to 25, 26 to 29) ..... (72 to 75, 76 to 79)
              DX1       DY1          DX2       DY2               DX7       DY7
```

Invisible vectors are specified by adding 500 to a positive X (but not Y) value and subtracting 500 from a negative X (but not Y) value. If the shape contains no lines, then the number of lines must be specified as zero and the coordinate records omitted. Brackets may be included in the line data to enable the data to be read more easily. For example, the line data for a shape 999 which is a square of 20 units inside a square of 30 units could be written as follows:

```
 999
S    10
          ( 510 -10)(   0  20)( -20   0)(   0 -20)(  20   0)( 505  -5)(   0  30)
          ( -30   0)(   0 -30)(  30   0)
```

The first record of the full sized text data contains an 'F' (column 1) and the number of full sized text strings associated with the shape (columns 6 to 8). Each subsequent record defines one of these strings in the following way:

(a) the X and Y absolute coordinates of the start position of the string (columns 12 to 15 and 16 to 19);

(b) the number of characters in the string (columns 21 to 25);

(c) the characters in the string, left justified (columns 31 to 46).

For example, the full sized text data to label the outer square of the shape previously defined might be:

```
                    F       1
                                ( -14 -14)    5     OUTER
```

The records for the half sized text are identical in format to those for the full sized text data. For example, the half sized text data to label the inner square of the shape 999 might be:

```
                    H       1
                                (  -9  -9)    5     INNER
```

The first record for the bond data contains a 'B' (column 1) and the number of bonds ssociated with the shape (columns 6 to 8). Each subsequent record provides up to four bond definitions and each bond definition provides the angle of the bond to the horizontal followed by the absolute coordinates of the bond start position. Valid bond angles range from 0° (horizontal, pointing right) to 345° in anticlockwise increments of 15° and each shape must have at least one bond. The format for bond records is as follows:

```
Columns: (12 to 15, 16 to 20, 21 to 24) (27 to 30, 31 to 35, 36 to 39)
           Angle 1      X1         Y1      Angle 2      X2         Y2

         (42 to 45, 46 to 50, 51 to 54) (57 to 60, 61 to 65, 66 to 69)
           Angle 3      X3         Y3      Angle 4      X4         Y4
```

For example, the bond data to add an outward pointing bond on each corner of shape 999 might be:

```
B     4
          (  45    10   10)( 135   -10   10)( 315    10 -10)( 225   -10 -10)
```

The data for a shape is terminated by a record having a space character in column 1 and this record should contain a shape number as the first item of data for the next shape.

A.2 Overcoming the problems of rotation

When a shape is rotated, the centre point of each string is rotated to shift the position of the string but the text remains horizontal, *ie* relative to the lines making up the shape, the text is rotated about the centre point of the string. The rotated relationship between the text and the lines might not be visually satisfactory (see Fig 4c) and interference can occur between the text and the lines (see Fig 4b). The user can take two steps to avoid the worst of these cases:

(a) define a *rotation relation* for the shape

(b) define all text strings with great care.

The first record for a shape can optionally contain an item of information to indicate how the shape should be rotated when the program finds it necessary to do so. Most shapes rotate satisfactorily by 180° to generate a mirror image but problems do occur when rotations of +90° and -90° are necessary. For such asymmetrical shapes, the program needs data for the shape as it should appear when rotated through 90° which can be used in place of the shape data for the unrotated shape. Such a related shape is known as a *rotation relation*. Columns 6 to 8 of the first record indicate whether a rotation relation exists and they can take one of the following three values:

(a) the number of a shape which is defined elsewhere in the data. For example, shapes 6 and 7 are rotation relations (see Fig 4a);

(b) a value of -1, meaning that the data definition of a rotation relation immediately follows the definition of the current shape. This rotation relation does not have a shape number, it cannot be accessed directly by the user's structure definition requests and it is only drawn when the current shape is to be rotated by +90° or -90°;

(c) a value of 0 or spaces indicating that there is no rotation relation and a strictly rotated form of the shape is always to be displayed.

Defining rotation relations caters for rotations of +90° and -90° when dealing with awkward shapes (see Fig 4b), but problems can occur even with relatively simple rotations of 180° if the text data is not carefully defined (see Fig 4c). In this example, the bonds should always point at the centre of the C atom but a strict rotation of 180° forces the bonds to point at the centre of the H atom. In order to overcome this problem, the 'C' must be defined as a separate string so that it rotates about its own centre and retains its position relative to the bond lines. In some way, the location of the 'H' must be defined relative to the position of the 'C'. If the number of characters in a string is specified as negative, the program forces the current string to retain the same position relative to the preceding string, whatever rotation is performed, *eg* the text data for the example given might be:

```
F     2
           (  4  10)   1    C
           ( 10  10)  -1    H
```

Note that the coordinate position of the 'H' is still expressed in absolute (unrotated) coordinates but it will always retain a position (+6, 0) relative to the start position of the string 'C', whatever rotation is performed, *ie* the string 'H' is attached to the string 'C'. An attached relationship can exist between several successive strings, even between the last full sized text string and the first half sized text string, but the first string for a shape cannot be an attached string since there is no preceding string to which it can be attached. Attached strings are always used for defining subscripts, *eg* for a shape containing 'CH_2', the text data might be 3

```
F      3
          (  0   0)    1     C
          (  6   0)   -1     H
          ( 12  -5)   -1     2
```

The use of attached strings to define subscripts can itself create unwanted side effects during rotation (see Fig 5). Here the problem is caused by the fact that no allowance is made for the presence of the subscript when the shape is rotated. This problem can be avoided by including a 'dummy' space character in the 'F' string so that the string is rotated about the approximate centre of the string 'F_2', rather than about the centre of the string 'F'.

```
F      8
          (  17  13)    2     F
          (  23 -18)   -1     2
          (  17 -29)    2     F
          (  23 -34)   -1     2
          ( -27 -13)    2     F
          ( -21 -18)   -1     2
          ( -27 -29)    2     F
          ( -21 -34)   -1     2
```

To be able to cope with a general rotation without interfering with line data or other character data, a string and any attached strings must be free to trace out a circle with a centre at the point of the rotation and the distance to the 'furthest' text point from the centre as the radius, *ie* in Fig 6 if the boxes represent characters, then no line data or other character data must intrude into the *circles if the shape is* to be rotated satisfactorily through all angles.

A.3 Examples of shape definition data

The following are some examples of shape definition data:

(1) The data listed below defines:

(a) Shape 63 which consists of 22 lines and has two bonds;

(b) Shape 64 which consists of the same lines as shape 63, and has two bonds;

(c) Shape 500 which is a dummy shape consisting of 10 bonds only, all positioned at point (0,0) and varying in angle from 0° to 315° in 45° increments to form an asterisk. This shape is used to check the appearance of other shapes when they are rotated.

The shape definition data for shapes 63, 64 and 500 (see Fig 7) is as follows:

```
  63   0
S     22
         (  16    0)(    9   15)(   -9   15)(    9   15)(   -9   15)( -16    0)(   -9 -15)
         (    9 -15)(   -9 -15)(    9 -15)( 500    4)(   -6   10)( 506   16)(   16    0)
         (-502    2)( -12    0)(-508   14)(    6   10)( 516    0)(    6 -10)( 500 -32)
         (   -6 -10)
F      0
H      0
B      2
         (    0  25    45)( 180   -9   45)
  64   0
B      2
         (    0  25    15)( 180   -9   45)
  500   0
 S      0
 F      0
 H      0
 B     10
          (180    0    0)(  90    0    0)( 270    0    0)(   0    0    0)
          (  0    0    0)( 180    0    0)( 135    0    0)( 315    0    0)
          ( 45    0    0)( 225    0    0)
```

(2) The data for shape 153 illustrates how a subscript can be defined as a string attached to the preceding string:

```
 153   0
 S     6
          (  15   -9)(   0 -16)( -15   -9)( -15   9)(    0  16)(  15   9)(
 F    11     -6     0    1    F
             17   -13    2    F
             23   -18   -1    2
             -2   -43    2    F
              4   -48   -1    2
             17   -29    2    F
             23   -34   -1    2
            -27   -13    2    F
            -21   -18   -1    2
            -27   -29    2    F
            -21   -34   -1    2
 H     0
 B     1
          (  90     0    0)
```

(3) The data for shape 1 illustrates the definition of a rotation relation immediately following the definition of the shape concerned:

```
     1  -1
 S       0
 F       3
              0    0    1    C
              6    0   -1    H
             12   -5   -1    3
 H       0
 B       1 (  90    2    9)
     0   0
 F       2
              0    0    2    CH
             12   -5   -1    3
 H       0
 B       1 ( 180   -2    4)
```

(4) The data for shapes 231, 279 and 280 illustrate how common data for similar shapes need only be defined once. These shapes share the same line and string data and only differ in the position of the bonds:

```
 231   0
S     12
         (  12   -7)(   0  -16)( -15   -9)( -15    9)(   0  16)(    12    7)(-501  -3)
         (  -7   -4)( 500  -16)(  10   -6)( 514    8)(   0  12)(
F      1
            -5   -2      1      N
H      0
B      2
         ( 180   -18   -7)(  0    12  -7)
 279   0
B      2
         ( 180   -18  -23)(  0    12  -7)
 280   0
B      2
         ( 180   -18   -7)(  0    12 -23)
```

(5) The data for shapes 466 and 416 illustrate how one shape can be defined as the rotation relation of another shape:

```
 466 416
S      0
F      7
            5    0     1     C
            0    0    -1     (
           11    0    -1     F
           17   -5    -1     2
           22    0    -1     )
           26   -6    -1     1
           32   -6    -1     6
H      0
B      2
         (  90    7     9)( 270    7  -2)(
 416 466
B      2
         (   0   22     4)( 180    3   4)
```

Appendix B

FORMAT OF THE STRUCTURE DEFINITION DATA

Each structure in the structure definition file is defined by a text descriptor record and by one or more chain records. Each record can be up to 80 characters in length.

Only the first 35 characters of the text descriptor record are used by the program. If less than 35 characters are supplied by the user, then the number is made up to 35 by the addition of spaces after the user supplied text. The first character of the text descriptor record must not be a hyphen so that it can be distinguished from a chain record.

Each chain record consists of a series of shape numbers separated by hyphens. The first character is a hyphen and the record contains no spaces. Whenever the shape is a link shape, an asterisk is included after the shape number and the number of asterisks indicates how many chains are to be connected to that link shape.

One chain description can extend over several consecutive records. A continuation request is indicated by terminating the record with a hyphen rather than a shape number. The following record is then considered as a continuation of the current record and no initial hyphen or shape number is required. For example, the record

```
-184-49-49-49
```

is equivalent to

```
-184-49-
49-49
```

The following paragraphs contain examples of structure definition data.

(1) In order to check the visual validity of the shape definition data, it is necessary:

(a) to display each shape in its normal orientation;

(b) to display each shape rotated through 90°, 180° and 270°;

(c) to display each shape rotated through intermediate angles

ie

45°, 135°, 225° and 315°.

The dummy shape 500 is used as a base shape to force the shape under consideration to be rotated (see Appendix A.3, Example 1). The data listed below could be used as a structure definition file to verify the shape data for shapes 63 and 64 which is given in Appendix A, Example 1. The output would be in the form of six drawings, three with title 'SHAPE 63' and three with title 'SHAPE 64', and this is shown in Fig 7.

The structure definition data is as follows:

```
 SHAPE  63
-63
 SHAPE  63
-500****
-500-63
-500-63
-500-63
-500-63
 SHAPE  63
-500********
-500
-500
-500
-500
-500-63
-500-63
-500-63
-500-63
 SHAPE  64
-64
 SHAPE  64
-500****
-500-64
-500-64
-500-64
-500-64
 SHAPE  64
-500********
-500
-500
-500
-500
-500-64
-500-64
-500-64
-500-64
```

(2) The data to produce Fig 8 is as follows:

```
GROUP 1663
-100*
-100-105*-7
-105-7

GROUP  1669
-100-102-51

GROUP  1670
-89-51-102

GROUP  1671
-100*
-100-105*-45
-105-45

GROUP  1673
-89-6-6

GROUP  1676
-64-102-31
```

(3) The data to produce Fig 11 is as follows:

```
POLYMER 162- 162- 162
-151**-151**-151**
-151-109
-151-109
-151-109
-151-109
-151-109
-151-109

POLYMER 162- 162- 441
-151**-151**-151**
-151-109
-151-109
-151-109
-151-109
-151-109
-151-125

POLYMER 162- 162- 824
-151**-151**-151**
-151-109
-151-109
-151-109
-151-109
-151-124
-151-124

POLYMER 162- 162- 637
-151**-151**-151**
-151-109
-151-109
-151-109
-151-109
-151-125
-151-125

POLYMER 162- 441- 441
-151**-151**-151**
-151-109
-151-109
-151-109
-151-125
-151-109
-151-125

POLYMER 162- 441- 824
-151**-151**-151**
-151-109
-151-109
-151-109
-151-125
-151-124
-151-124
```

Appendix C

FEATURES OF THE GRAPHICS SOFTWARE

The FORTRAN programs make use of the following features of the graphics software package, GPGS:

(a) the screen is cleared by a call to the routine CLRDEV;

(b) picture entities are declared between calls to the routines BGNPIC and ENDPIC and each picture is given a unique integer identifier;

(c) lines are drawn by a routine LINE with parameters:

X-coordinate
Y-coordinate
Pen state (up or down)

(d) an integer identifier can be associated with a line, a collection of lines or with a string of characters which are declared between calls to the routines BGNNAM and ENDNAM;

(e) a routine INWAIT will wait for a user response on a given range of devices, the keyboard or light pen in this case, and it provides the following information about the response: (i) an integer identifier specifying which device responded;

(ii) the data provided by the device, *ie* for a light pen, the picture identifier and the line/character string identifier for the item selected; or for the keyboard, the characters typed by the user;

(f) the limits for the drawing space are declared in a call to the routine WINDW and this coordinate space is mapped onto the screen area. All parameters subsequently passed to the line drawing routines must be expressed in terms of coordinates, within the declared drawing space.

Appendix D

A DESCRIPTION OF THE STRUCTURE DISPLAY PROGRAM IN EXECUTION

This Appendix contains a description of the sequential operations performed by the structure display program in displaying a chemical structure. The principal routines in the program and their functions are:

GETSH (GET SHape) obtains the next shape number;
TOSTAR (TO STARt position) matches bonds and calculates the shape origin and rotation;
DRAWSH (DRAW SHape) draws the shape.

The program first enters an initialisation phase as described in section 8.1 in order to establish the working environment. It then opens the random access file RANDAT containing the shape definitions and reads into memory the shape index, which defines where each shape description is stored within the file.

The routine GETSH is entered to obtain from the structure description data the number of the first shape to be drawn and the number of asterisks associated with the shape number, ie it remembers if the shape is a link shape and how many chains are to be attached to it. Finally it reads the shape description for the shape number.

The routine TOSTAR is entered to establish whether the first shape can be displayed without rotation. The shape is examined for bonds at 0^o and then for bonds at 180^o, 270^o and 90^o successively. If such a bond is found, the shape can be drawn in its normal orientation and this bond will be used as the exit bond to connect to the next shape. Thus the order of priorities in asserting the original direction of drawing are:

right horizontal
left horizontal
down vertical
up vertical.

If no horizontal or vertical bond is found, the normal shape connection method (see section 6) is used to select an exit bond to match with a fictitious 180^o bond and the shape is thus rotated to transform the selected bond into a bond at 0^o. This forces the first chain to start in a right, horizontal direction. A bond opposite the selected exit bond is taken as the assumed entry bond. If the overlap check has been requested, TOSTAR calls the routine DRAWSH in 'non-draw' mode to establish the maximum X and Y coordinates of the first shape.

The routine DRAWSH is entered in 'draw' mode to draw the shape with any required rotation. It takes the current shape description and draws the appropriate lines, full sized characters and half sized characters relative to the current origin, transforming all coordinate positions according to the current rotation. It then adds bonds of a standard length at the transformed coordinate positions and at the transformed angles, and each bond is given a unique light pen identity, which is for use in the interaction phase. Having drawn the shape the processing of the first shape is completed.

The routine GETSH is entered again. If the shape just drawn was a link shape then GETSH updates the 'future link shape' stack with

(a) its shape number

(b) the X and Y coordinates of the current origin

(c) the current rotation which has been applied to the shape

(d) the number of link chains to be processed

(e) the light pen identity given to the first bond of the shape

(f) all the bond angles for the shape except

(i) that bond used for entry to the shape, or, for the first shape of a structure, the bond which has been assumed to be the entry bond;

(ii) one bond opposite (180^o out of phase with) the entry bond or assumed entry bond, where the shape just drawn was the last shape in a chain.

The stack operates on a 'first in - first out' basis, and all operations on the stack are performed by GETSH. In addition, a flag is set to indicate that the next exit/entry connection to be made will involve a link shape and so the 'future link shape' stack of available bonds must be updated on the next entry to GETSH to delete the exit bond. GETSH next repeats the process whereby it isolates the next shape number, records the number of asterisks and reads the shape description into the current shape description area. The program now has one shape completely drawn and all the information necessary to draw the next.

The routine TOSTAR decides how the two shapes should be connected together and it implements the rules described in section 6. Having established which bonds are to be linked together, it takes the origin position for the previous shape and any rotation associated with it and it calculates a new origin and a new rotation so that when the shape is drawn with these parameters, the required bonds will connect. If a rotation of $+90^o$ or -90^o is required and a rotation relation exists, the current shape definition is overwritten with that of the rotation relation and the whole matching process is repeated. If the interference check has been requested, the routine GETLEN is entered to find the rectangle containing the current shape and to check for overlaps between this rectangle and the rectangle containing the previous shape. If the rectangles overlap, the entry bond is first shortened to zero, and then lengthened up to 14 times the standard bond length in an attempt to avoid the overlap.

The routine DRAWSH draws the shape as before.

This whole process is repeated until GETSH completes the processing of a chain record and any associated continuation records. At this stage, if the 'future link shape' stack is empty then the structure is fully drawn. If it is not empty, the information on the top of the stack is accessed to restore the program state to that current when the link shape was first drawn, and the link shape flag is set so that after TOSTAR has selected an exit bond for the link shape and on the next entry to GETSH this bond will be deleted from those stored on the 'future link shape' stack. On the same pass, the number of link chains to be processed is decremented. A link shape stays on top of the stack until this count reaches zero, when the following item then becomes the new 'top of stack' and indicates the next link shape to be used. Having reverted to an earlier link shape as the 'previous' shape, the drawing logic is then repeated as before.

When a structure has been drawn completely, the program calculates the offset which must be applied to centralise the rectangle containing the drawing and this offset is applied if the structure is displayed again during the interaction phase or if a copy file is produced.

After the required array of structures has been drawn, the interaction phase is entered. A light pen selection of a bond indicates that the user wishes to change the length of that bond. The graphics display driver returns the unique light pen identity of the bond selected and the program appends it to a list, together with the length of the bond required, which is read from the Vector General keyboard. This list is accessed by DRAWSH. As it draws each bond it checks whether this bond has been selected during this program run and if so, it uses the requested bond length stored in the list; otherwise it uses the standard bond length. The list is also used by TOSTAR when reverting to a previously stored link shape so that it can calculate the correct shift of origin to connect two selective bonds together. Following each light pen interaction, the whole array of structures is redrawn and this continues until the user responds with a RETURN on the Vector General keyboard to indicate that the drawing is now acceptable. Any light pen selection of a menu item is also processed at this time.

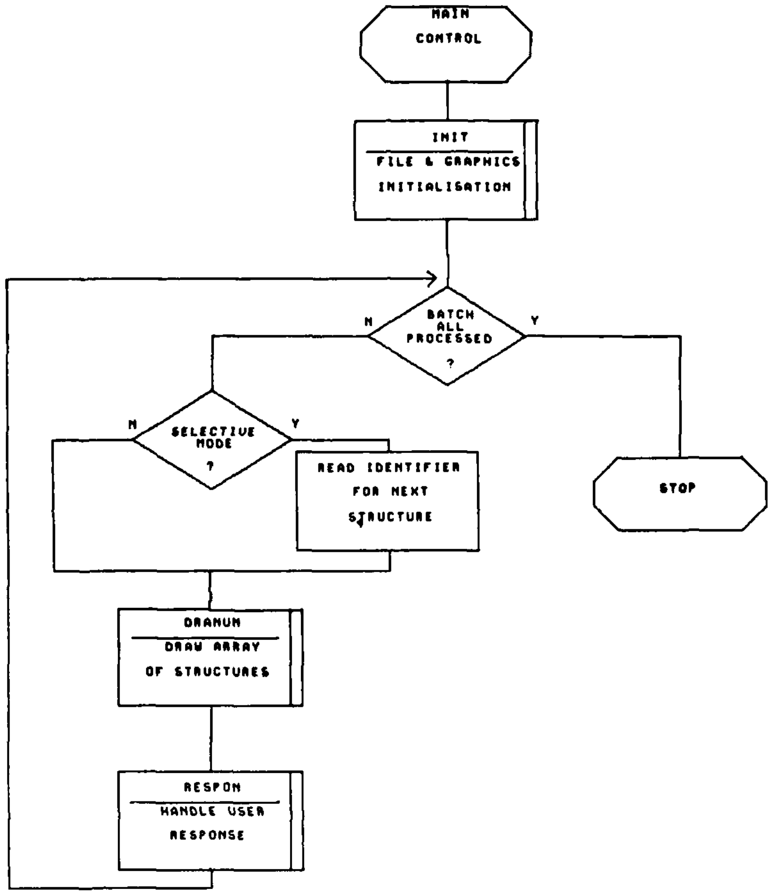
If a copy file has been requested the program makes a further pass through the data to produce the final, centred drawing data which can later be processed to produce drawings on a hard copy device. After the copy has been produced the program either loops to draw the next array or terminates if the structure description file is exhausted.

Because the program may make more than one pass through the structure description data and it may make many identical accesses to the shape description data, considerable buffering of data takes place. The structure description data for the complete array of structures is maintained within the program and it is only read from the file on the first pass. The shape description data is also buffered, but in a cyclic buffer. If a shape needs to be drawn and the description has already been read into the buffer, the shape data is advanced round the buffer in order to delay the time when it is overwritten by a new, incoming shape description. The descriptions of shapes which are in current common usage thus tend to stay in the buffer and are immediately available.

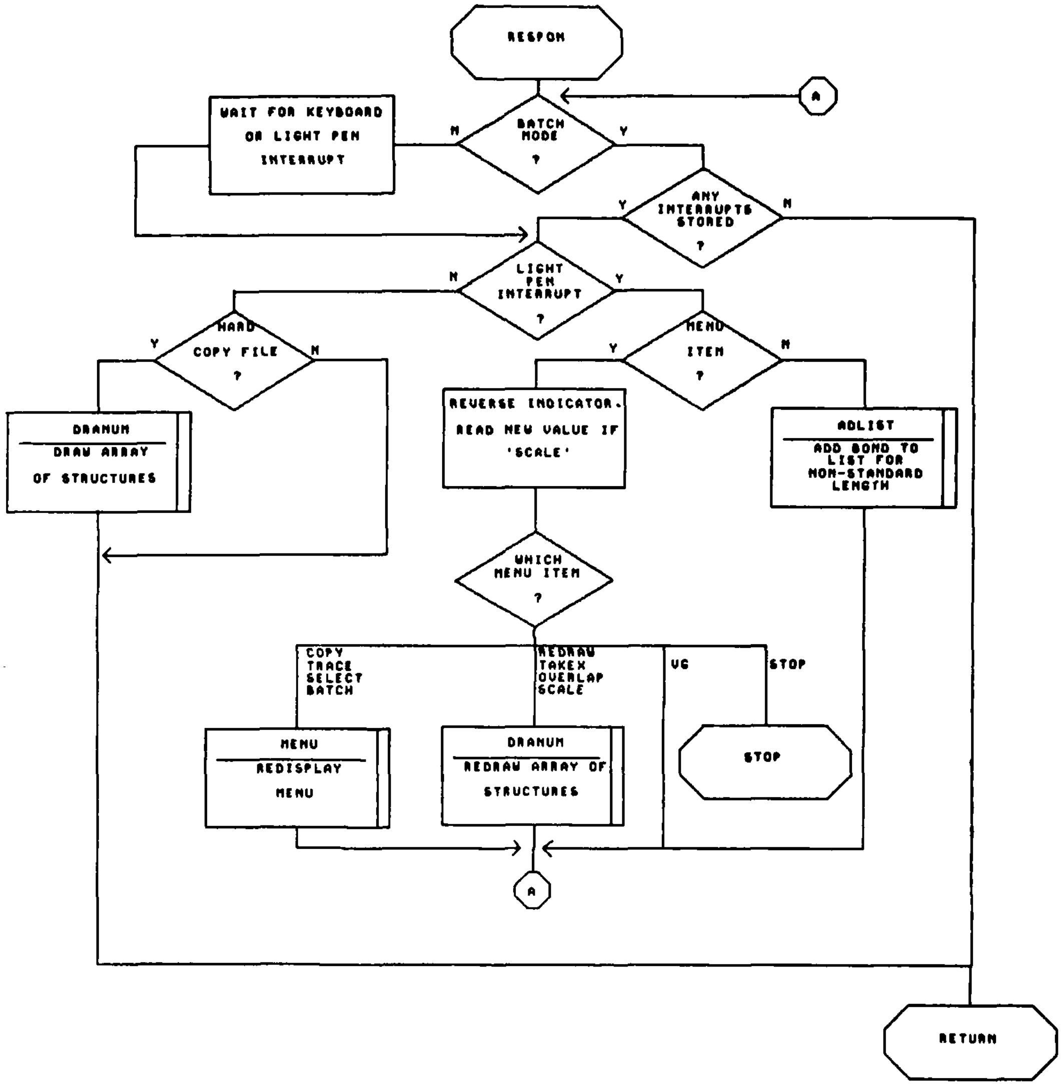
The logic of the program is shown in flowchart form in Appendix E.

Appendix E

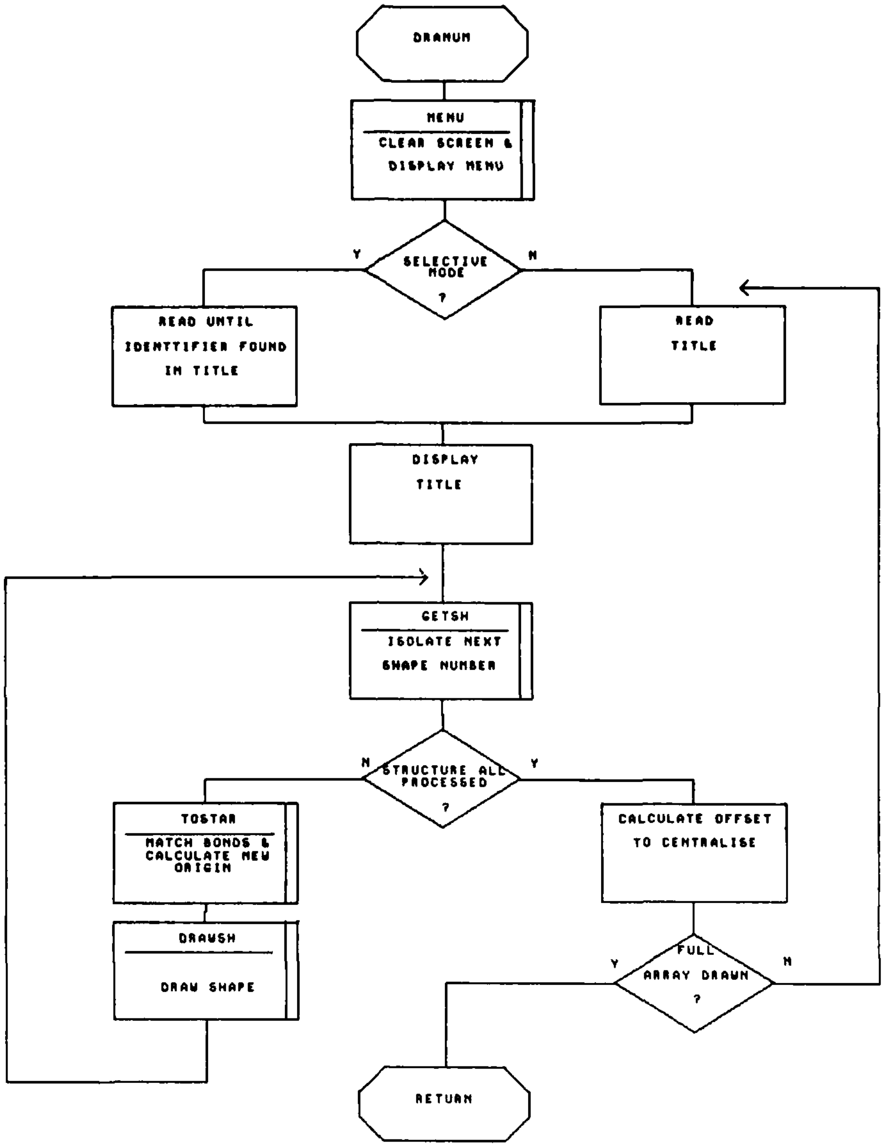
STRUCTURE DISPLAY PROGRAM FLOWCHARTS



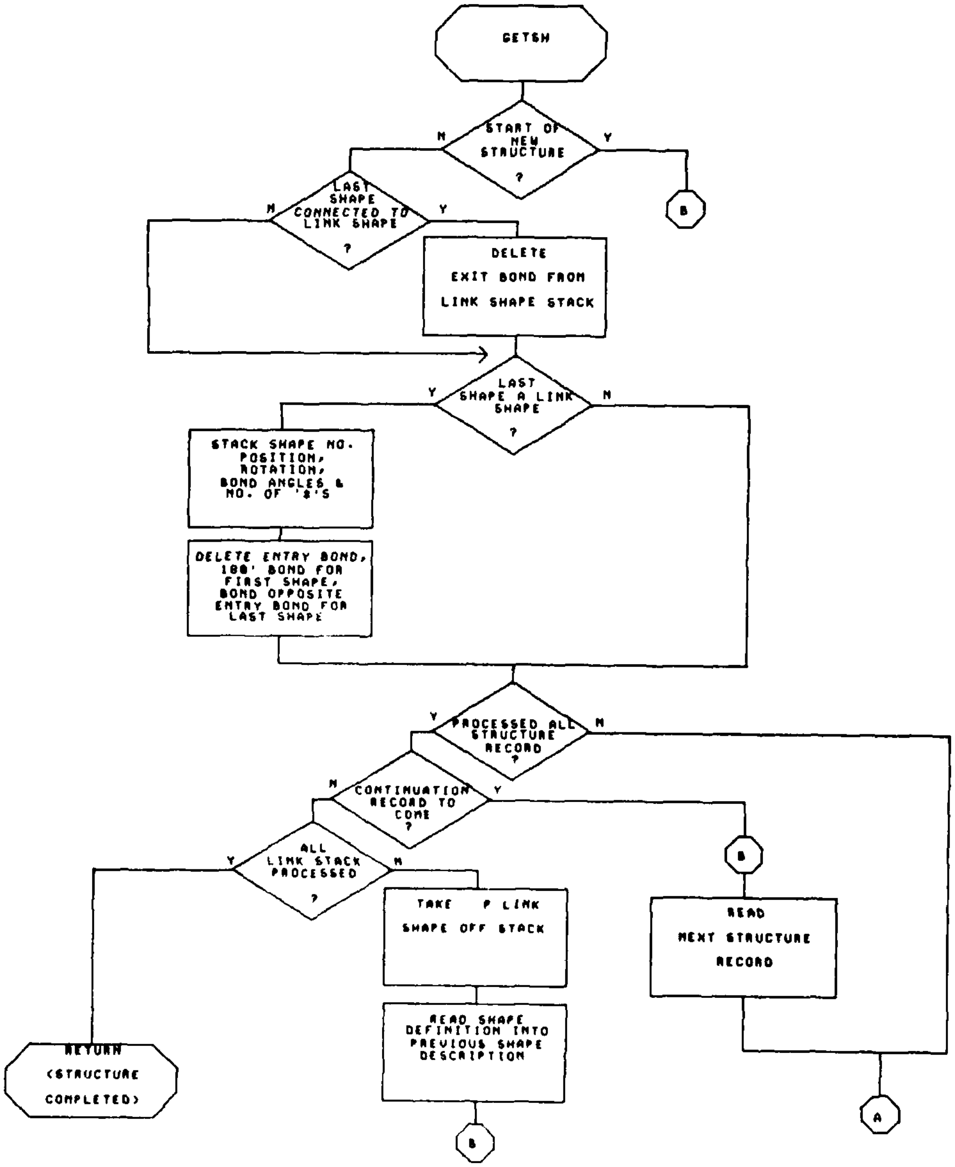
Main control routine



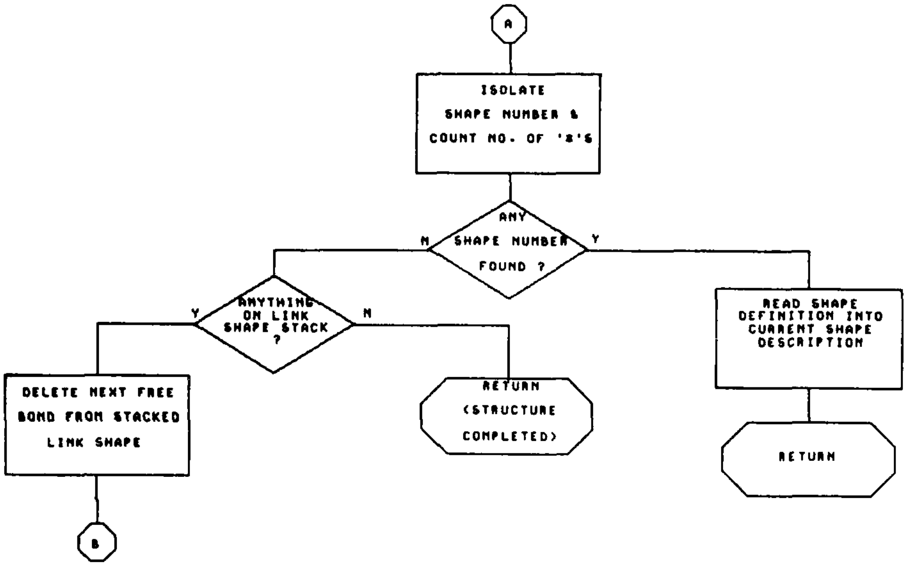
RESPON: Routine to handle user interactions



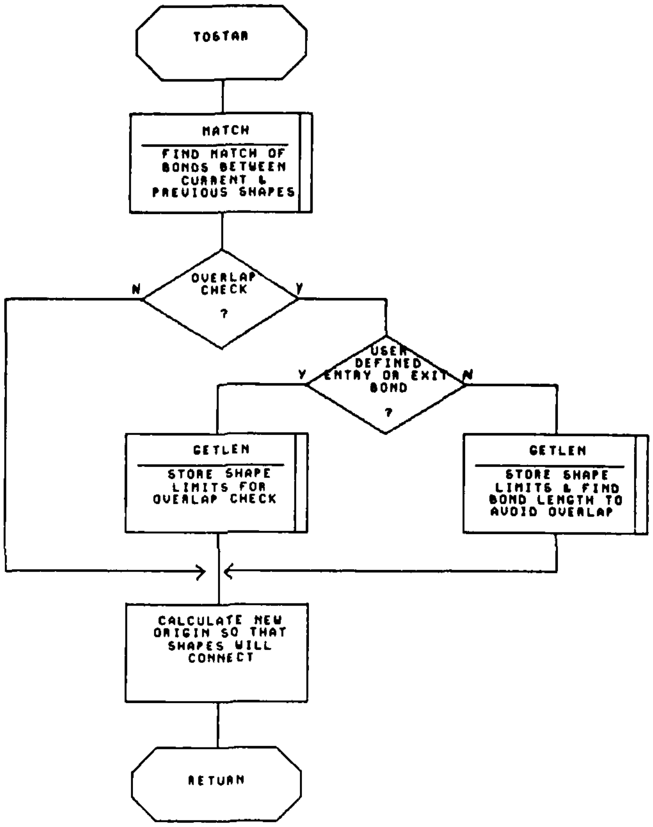
DRANUM: Routine to draw an array of structures


GETSH
START OF NEW STRUCTURE ?
N
Y
B
LAST SHAPE CONNECTED TO LINK SHAPE ?
N
Y
DELETE EXIT BOND FROM LINK SHAPE STACK
LAST SHAPE A LINK SHAPE ?
Y
N
STACK SHAPE NO. POSITION, ROTATION, BOND ANGLES & NO. OF '$'S
DELETE ENTRY BOND, 180' BOND FOR FIRST SHAPE, BOND OPPOSITE ENTRY BOND FOR LAST SHAPE
PROCESSED ALL STRUCTURE RECORD ?
Y
N
CONTINUATION RECORD TO COME ?
N
Y
ALL LINK STACK PROCESSED ?
Y
N
B
TAKE P LINK SHAPE OFF STACK
READ NEXT STRUCTURE RECORD
READ SHAPE DEFINITION INTO PREVIOUS SHAPE DESCRIPTION
RETURN (STRUCTURE COMPLETED)
B
A

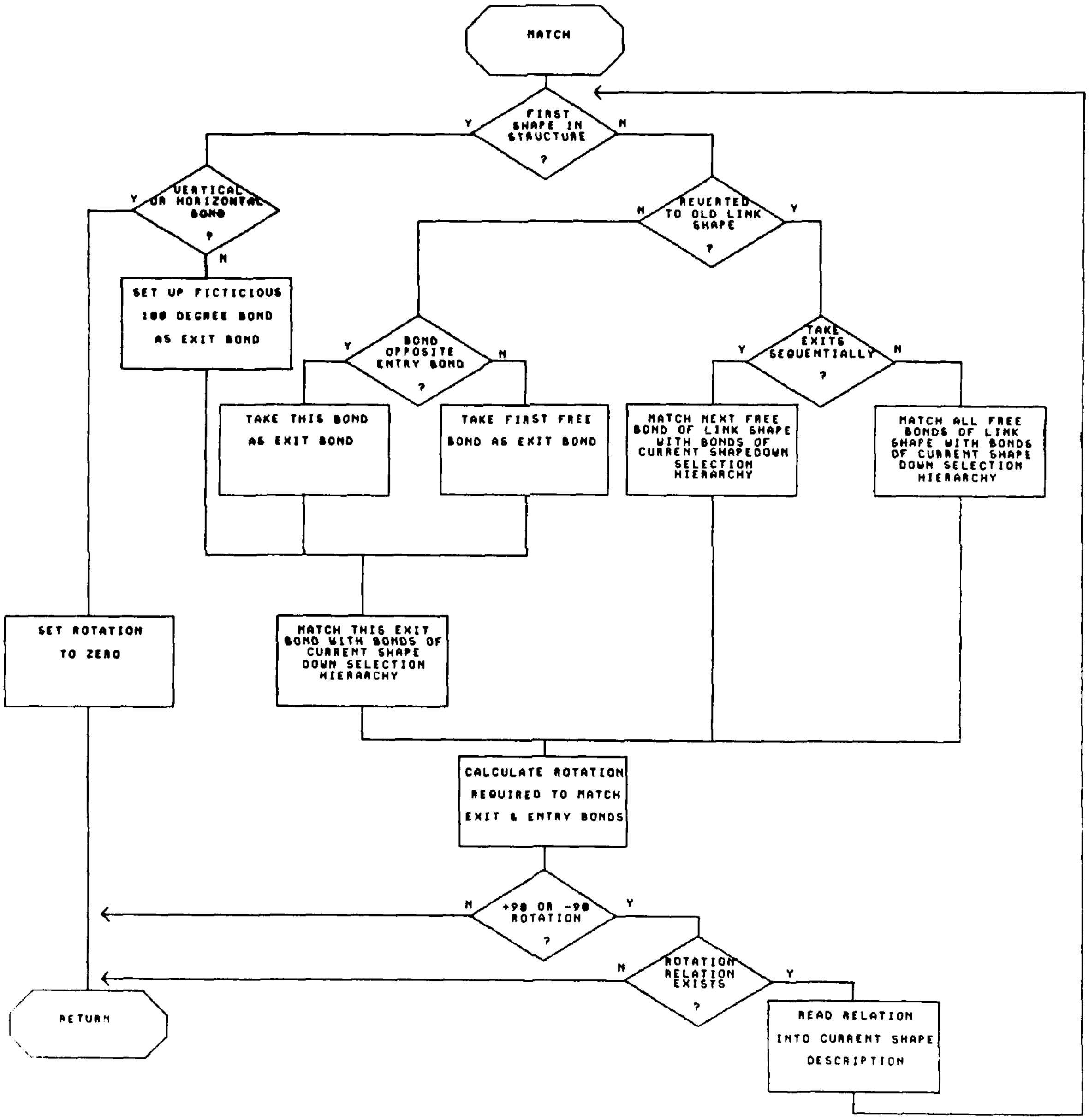
GETSH (Part 1): Routine to isolate the next shape number


A
ISOLATE SHAPE NUMBER & COUNT NO. OF '*'S
ANY SHAPE NUMBER FOUND ?
N
Y
ANYTHING ON LINK SHAPE STACK ?
Y
N
READ SHAPE DEFINITION INTO CURRENT SHAPE DESCRIPTION
DELETE NEXT FREE BOND FROM STACKED LINK SHAPE
RETURN (STRUCTURE COMPLETED)
RETURN
B

GETSH (Part 2): Routine to isolate the next shape number


TOSTAR
MATCH
FIND MATCH OF BONDS BETWEEN CURRENT & PREVIOUS SHAPES
OVERLAP CHECK ?
N
Y
USER DEFINED ENTRY OR EXIT BOND ?
Y
N
GETLEN
STORE SHAPE LIMITS FOR OVERLAP CHECK
GETLEN
STORE SHAPE LIMITS & FIND BOND LENGTH TO AVOID OVERLAP
CALCULATE NEW ORIGIN SO THAT SHAPES WILL CONNECT
RETURN

TOSTAR: Routine to calculate the new shape origin and rotation


MATCH
FIRST SHAPE IN STRUCTURE ?
Y
N
VERTICAL OR HORIZONTAL BOND ?
Y
N
SET UP FICTICIOUS 180 DEGREE BOND AS EXIT BOND
REVERTED TO OLD LINK SHAPE ?
N
Y
BOND OPPOSITE ENTRY BOND ?
Y
N
TAKE THIS BOND AS EXIT BOND
TAKE FIRST FREE BOND AS EXIT BOND
TAKE EXITS SEQUENTIALLY ?
Y
N
MATCH NEXT FREE BOND OF LINK SHAPE WITH BONDS OF CURRENT SHAPEDOWN SELECTION HIERARCHY
MATCH ALL FREE BONDS OF LINK SHAPE WITH BONDS OF CURRENT SHAPE DOWN SELECTION HIERARCHY
SET ROTATION TO ZERO
MATCH THIS EXIT BOND WITH BONDS OF CURRENT SHAPE DOWN SELECTION HIERARCHY
CALCULATE ROTATION REQUIRED TO MATCH EXIT & ENTRY BONDS
+90 OR -90 ROTATION ?
N
Y
ROTATION RELATION EXISTS ?
N
Y
READ RELATION INTO CURRENT SHAPE DESCRIPTION
RETURN

MATCH: Routine to select the bonds of the current and previous shapes to be connected

Appendix F

EXECUTION EXCEPTION CONDITIONS

F.1 The shape definition program

The shape definition program checks the data as it is read in and whenever the definition rules are violated an error message is printed out. The messages take the following form

PPP NNN : Message

where 'PPP' is the previous shape number
'NNN' is the current shape number, or 0 for the rotation relation of PPP,
and 'Message' indicates what type of discrepancy has been discovered.

After printing the message, the program continues as if no error had been found.

The possible messages and their causes are:

(a) INVALID SHAPE NUMBER : shape number greater than 500

(b) MORE THAN 83 LINES : the maximum number of lines has been exceeded

(c) MORE THAN 15 FULL SIZED STRINGS : the maximum number of full sized strings has been exceeded

(d) INVALID FULL SIZED STRING LENGTH: a full sized string has been defined with no characters or with more than 5 characters

(e) MORE THAN 2 HALF SIZED STRINGS : the maximum number of half sized text strings has been exceeded

(f) INVALID HALF SIZED STRING LENGTH: a half sized string has been defined with no characters or with more than 15 characters

(g) INVALID NUMBER OF BONDS : the maximum number of bonds has been exceeded

(h) MTH BOND ANGLE INVALID:RRR : bond angle number M has value RRR which is not a multiple of 15°

(i) TOO MANY SHAPES : the maximum number of shapes has been exceeded

(j) INVALID FIRST CHARACTER : the first character of a record is not '∇', 'F', 'S', 'H' or 'B'. The character is assumed to be a space

F.2 The structure display program

The program recognises the following faults in processing the input data and produces a message where appropriate:

(a) Blank data records in the structure definition file are ignored.

(b) PAUSE:UNKNOWN CHARACTER: The program has encountered an unexpected character when processing a chain description record, *ie* not "0" to "9", '-' or '*'. The character is ignored if the program is restarted.

(c) PAUSE: NUMBER UNTERMINATED: A field of more than ' characters has been found when analysing the next shape number. This field width includes digits and '*'s. The number is assumed to be terminated by the 15th character if the program is restarted.

(d) NO BOND MATCH POSSIBLE: It is not possible to link the current shape to the previous shape because

(i) a non-link shape has only one bond and so no bond is available as an exit bond;

(ii) a link shape has no free bonds left to be used as an exit bond.

A blinking '##' is displayed on the Vector General screen as a further warning to the user.

(e) ***SHAPE NOT DEFINED NNN***: This message is displayed when a shape requested in the structure definition file has not been defined in the shape definition file. The invalid shape number is ignored and, arbitrarily, shape 500 is used instead.

(f) TOO MANY BONDS ALTERED: The user has attempted to define the length of more bonds than the program can store. The program ignores the current request to change a bond length.

(g) TOO MANY STRUCTURE DEFINITION RECORDS: The number of records or the number of characters defining the current array of structures has exceeded the maximum which the program can store. Program execution is terminated.

(h) TOO MANY LINK SHAPES STACKED: The number of link shapes stored on the stack for future use has exceeded the maximum size of the stack. Program execution is terminated.

(i) NO. OF OVERLAP BOXES EXCEEDED: The number of shapes in the current structure has exceeded the limit which can be handled by the overlap check. Execution is continued and the limits of the first rectangle are overwritten by the limits for the current shape.

Table 1

ORDER OF SEARCH FOR A MATCH BETWEEN THE BONDS OF A LINK SHAPE AND THE CURRENT SHAPE

	Link bond 1		Link bond 2		Link bond 3	
	Current bond 1	Current bond 2	Current bond 1	Current bond 2	Current bond 1	Current bond 2
PHASE DIFFERENCE deg						
180	1	2	9	10	17	18
0	3	4	11	12	19	20
90	5	6	13	14	21	22
270	7	8	15	16	23	24
210	25	26	33	34	41	42
30	27	28	35	36	43	44
120	29	30	37	38	45	46
300	31	32	39	40	47	48
225	49	50	57	58	65	66
45	51	52	59	60	67	68
135	53	54	61	62	69	70
315	55	56	63	64	71	72
240	73	74	81	82	89	90
60	75	76	83	84	91	92
150	77	78	85	86	93	94
330	79	80	87	88	95	96

REFERENCES

No.	Author	Title, etc
1	-	Introduction to RSX-11M. Digital Equipment Corporation, AA-2555C-TC
2	-	GPGS Reference Manual. Technische Hogeschool, Delft, RC-GFX-75002-A
3	-	GPGS Users Tutorial. Technische Hogeschool, Delft, RC-GFX-75003-0

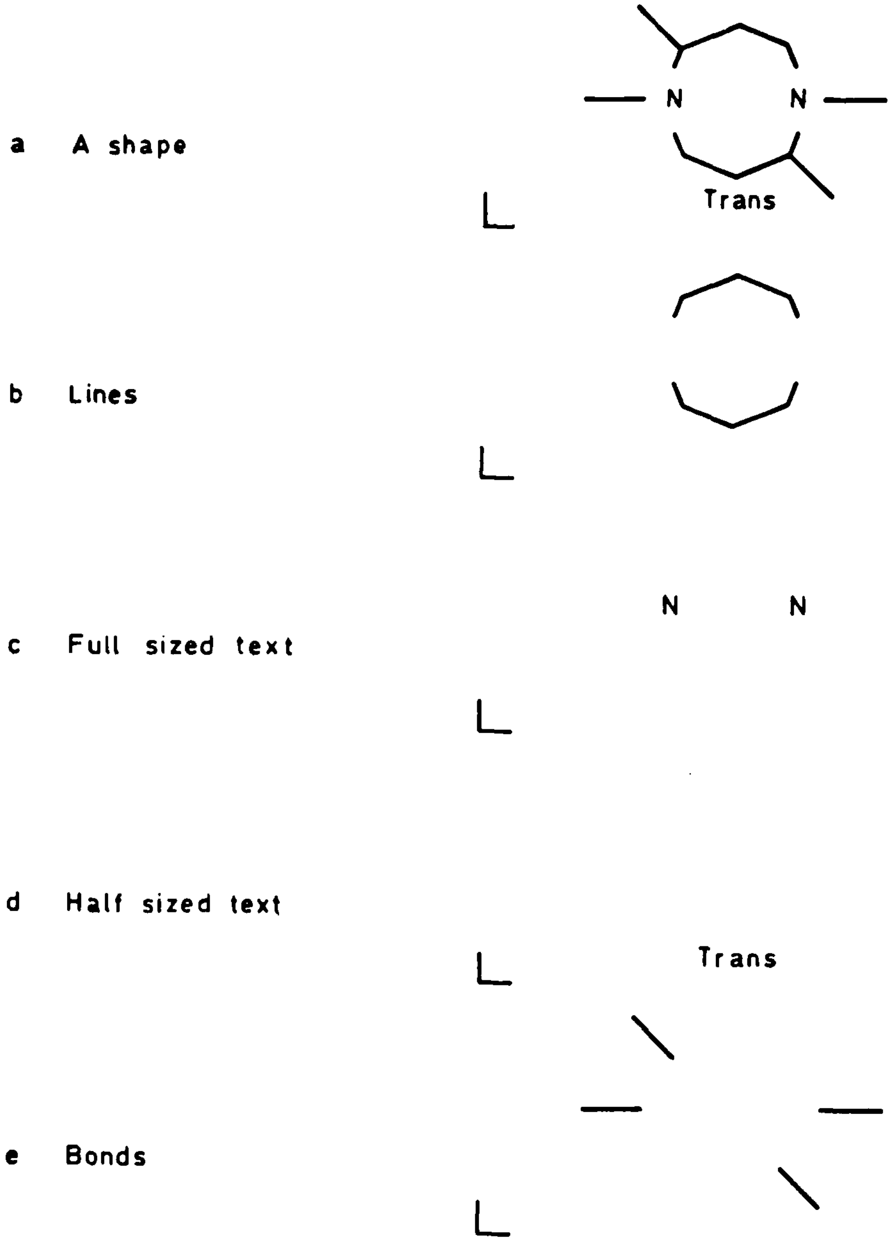
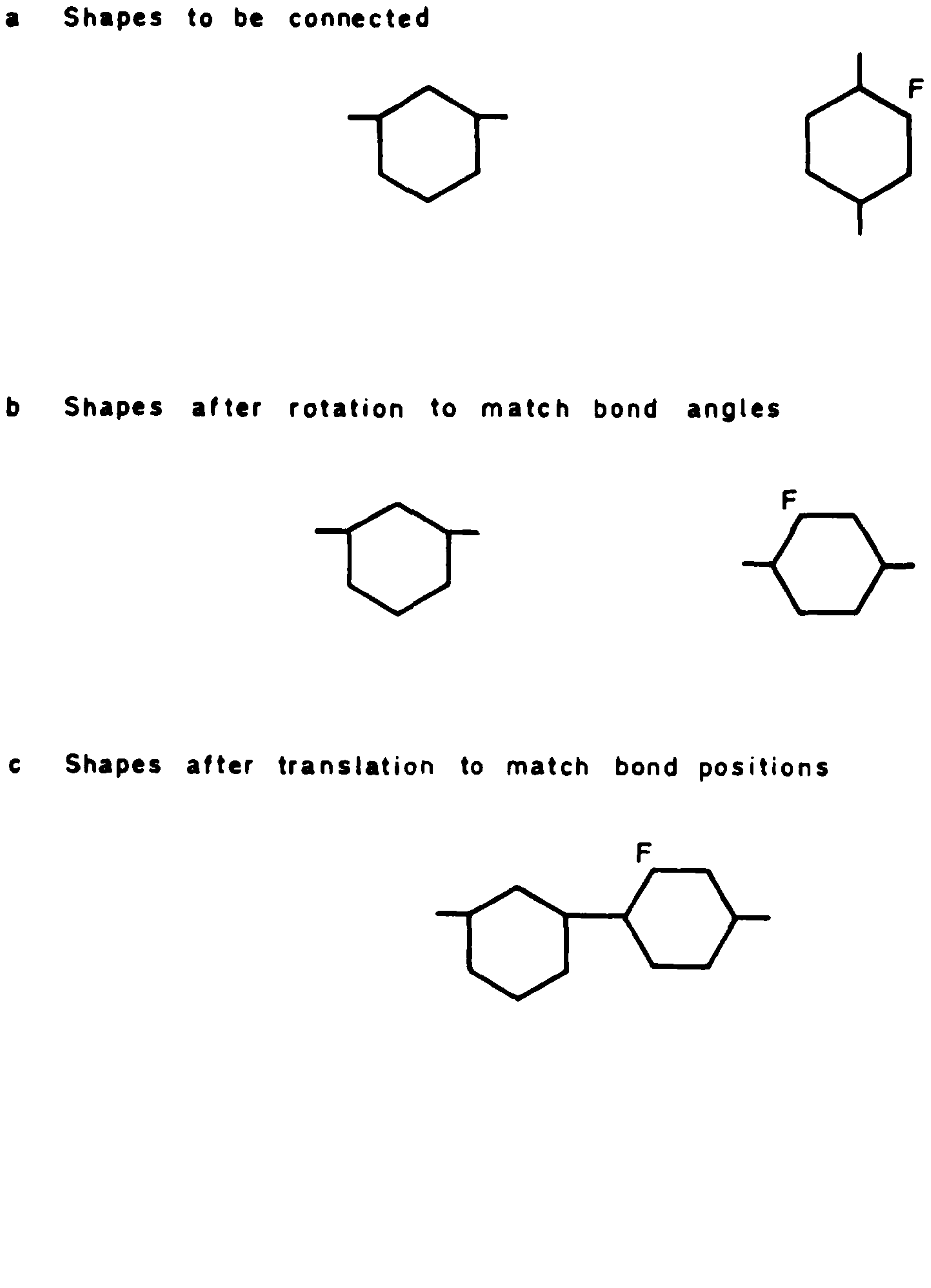


Fig 1a-e A shape and its graphical constituents

Fig 2a-c

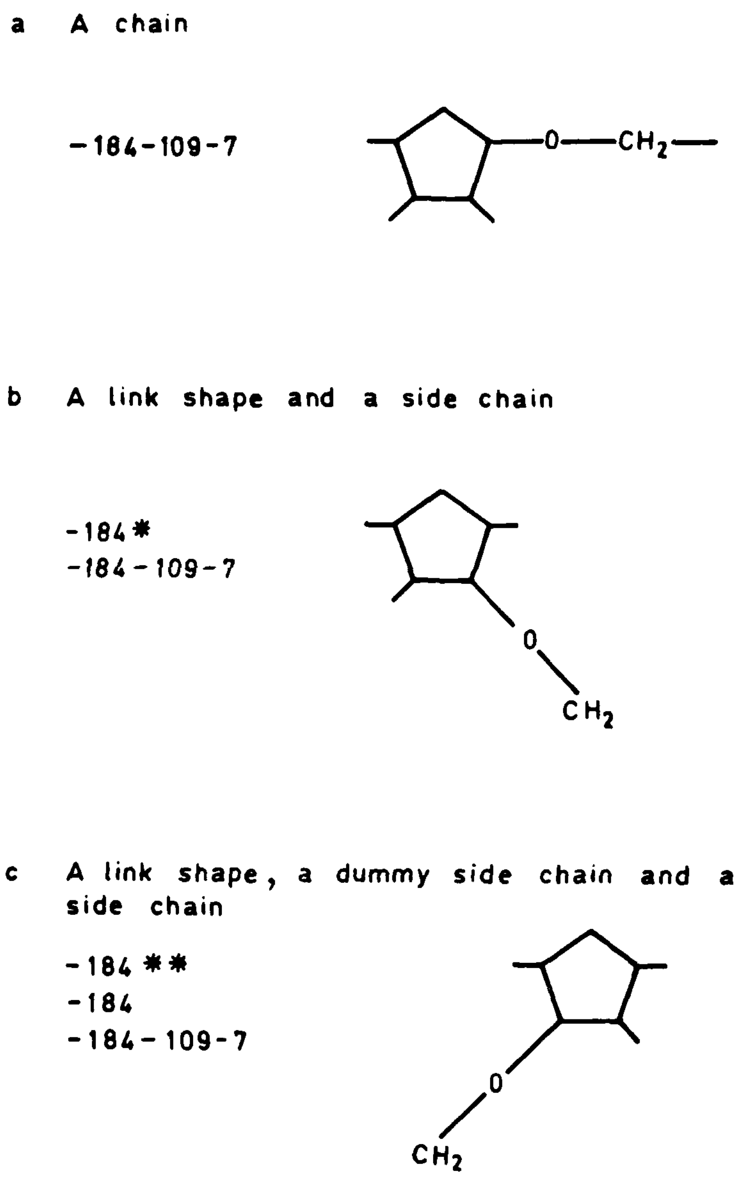


a Shapes to be connected

b Shapes after rotation to match bond angles

c Shapes after translation to match bond positions

Fig 2a-c Rotation and translation of shapes in order to connect them togethe.



a A chain

-184-109-7

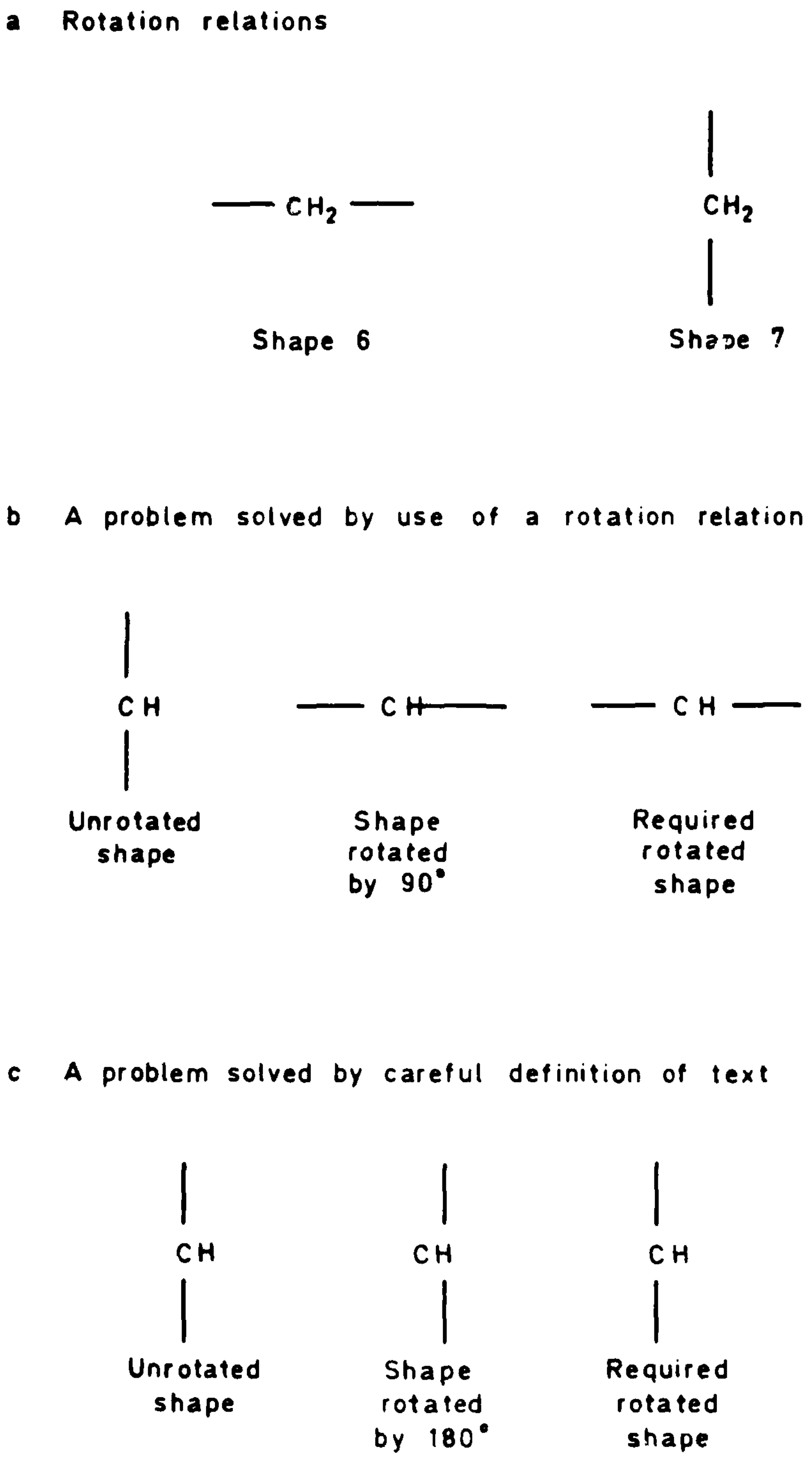
b A link shape and a side chain

-184*
-184-109-7

c A link shape, a dummy side chain and a side chain

-184**
-184
-184-109-7

Fig 3a-c Examples of the use of link shapes and dummy side chains

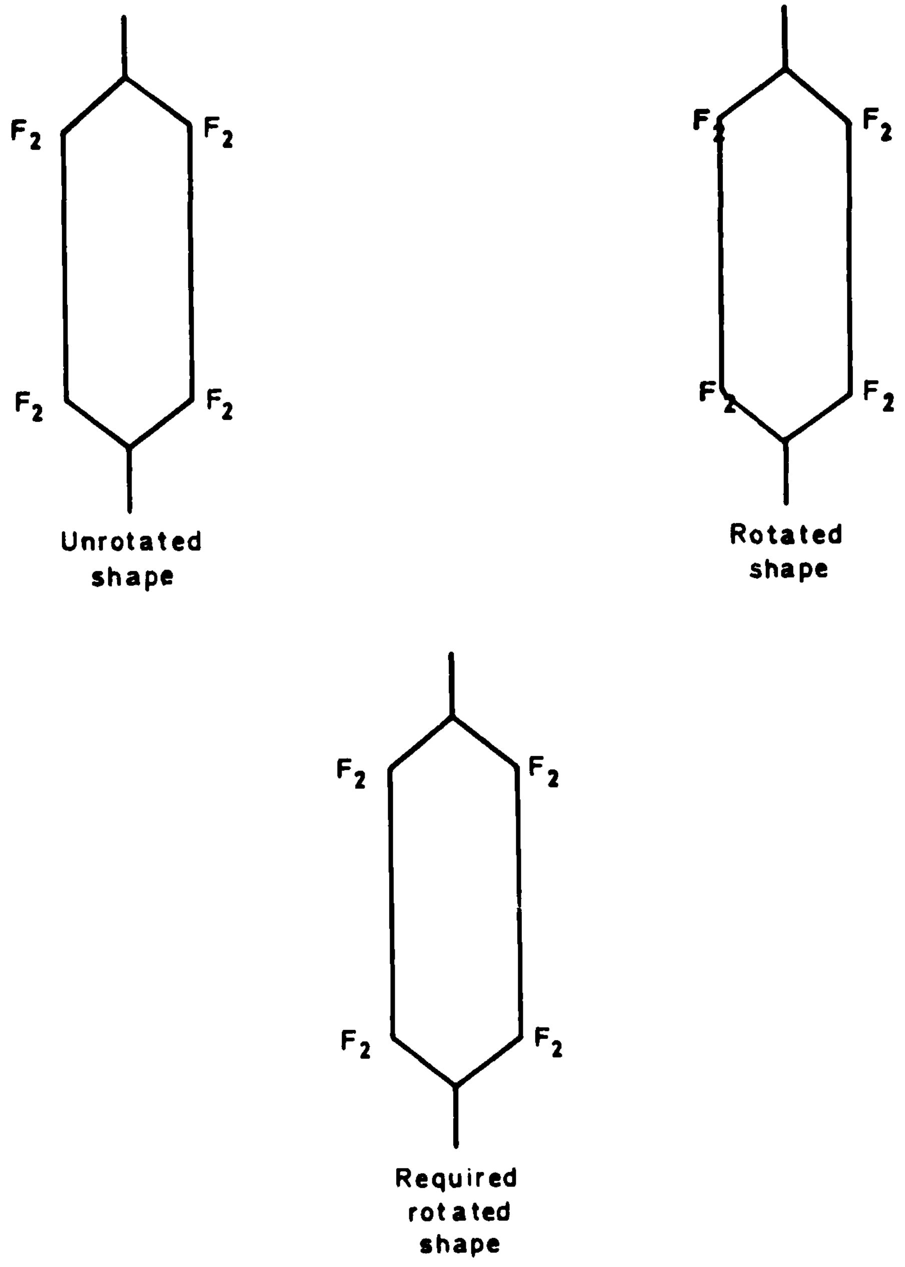


a Rotation relations

b A problem solved by use of a rotation relation

c A problem solved by careful definition of text

Fig 4a-c Problems of shape rotation



Unrotated shape

Rotated shape

Required rotated shape

Fig 5 Problems of rotating subscript text

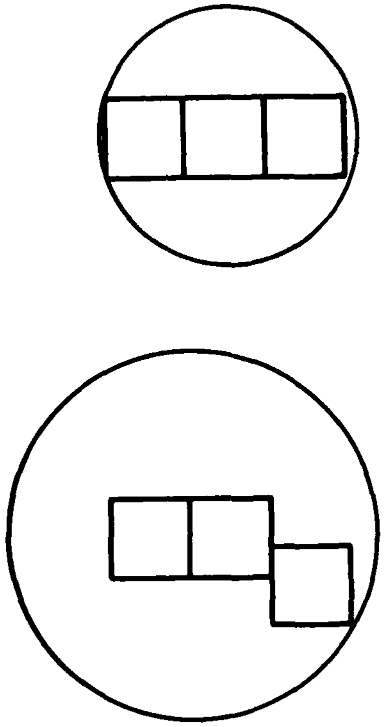


Fig 6   Clearance area to allow for character rotation

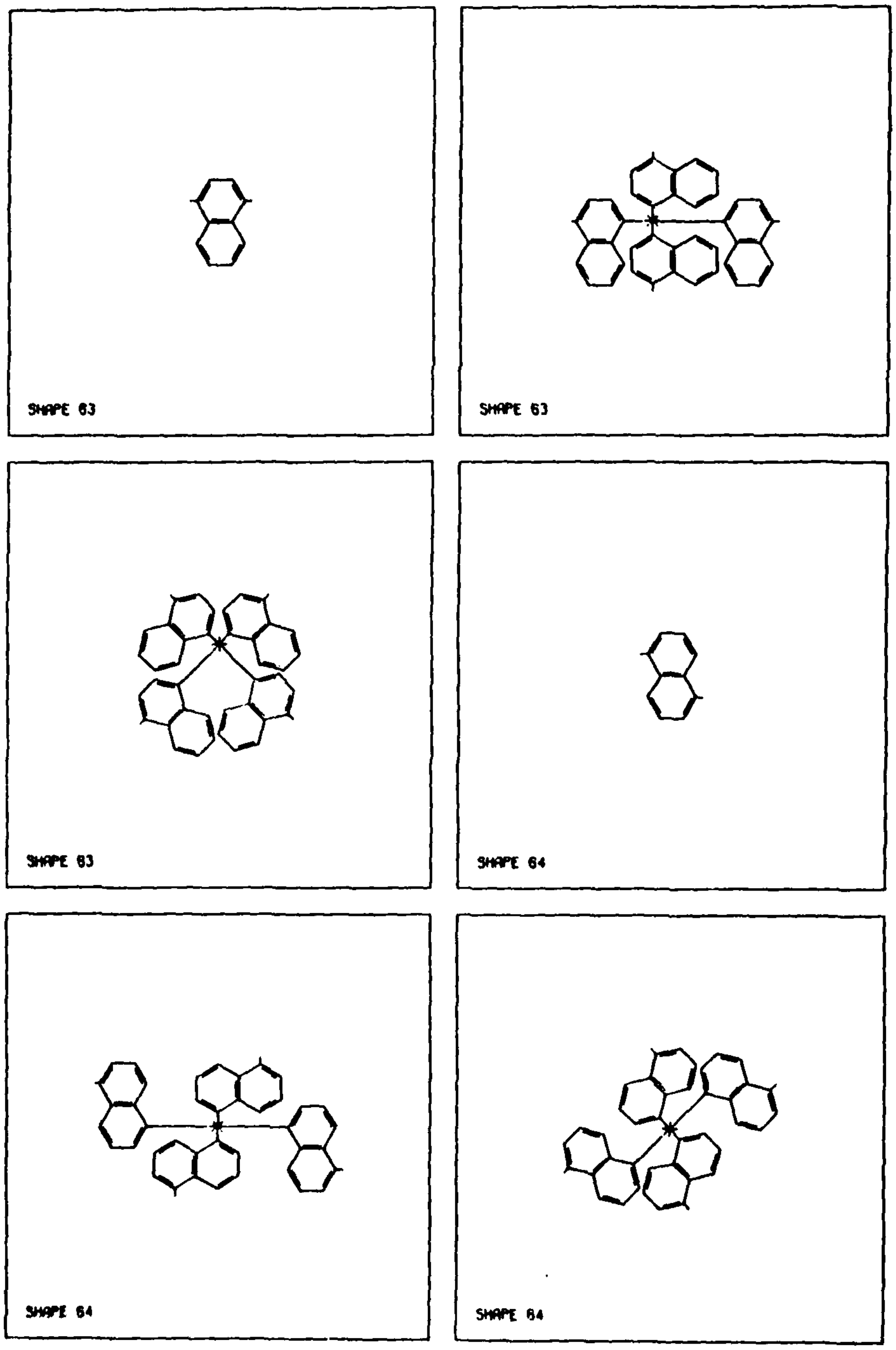


Fig 7 Shapes displayed in various orientations

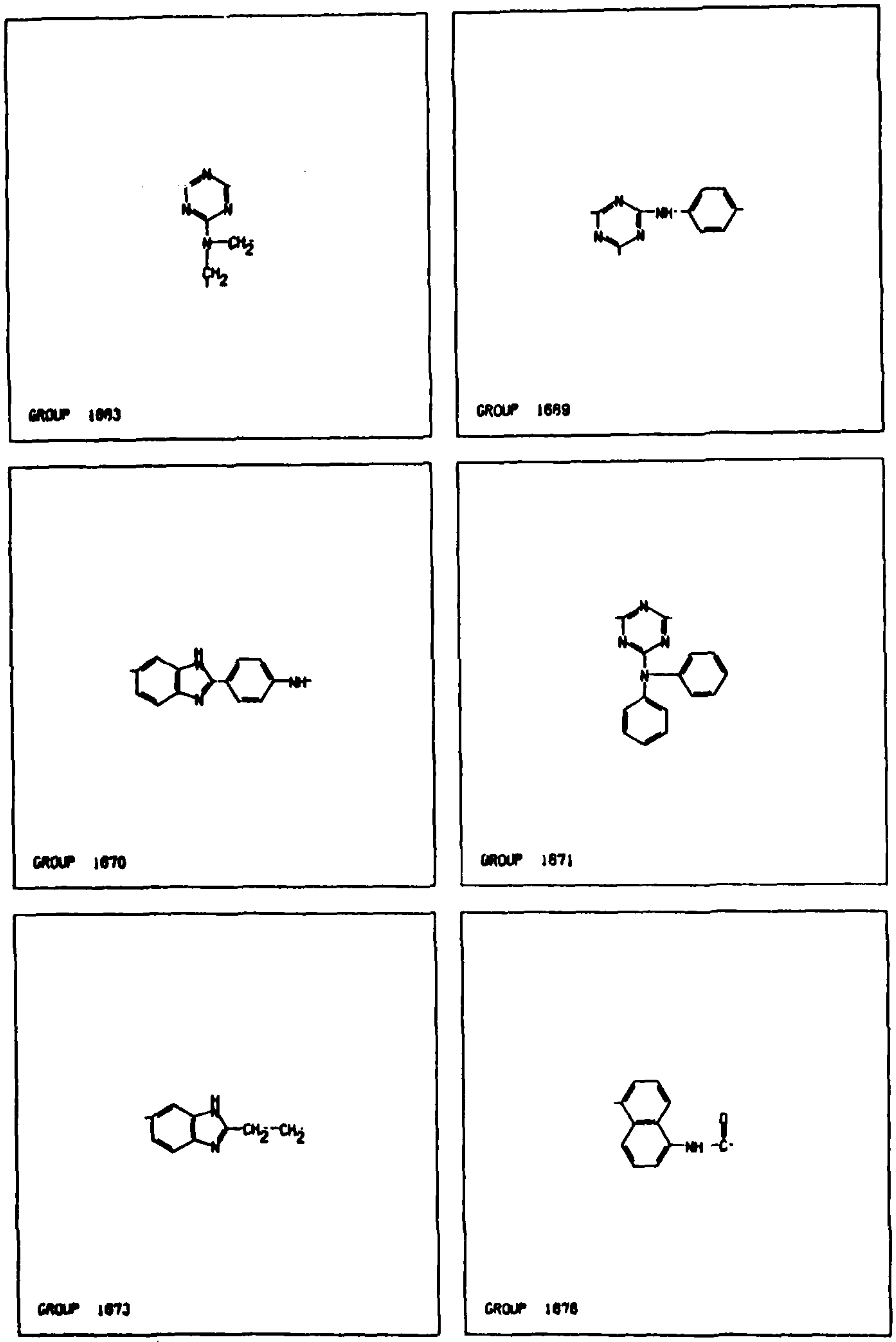


Fig 8 Examples of group drawings

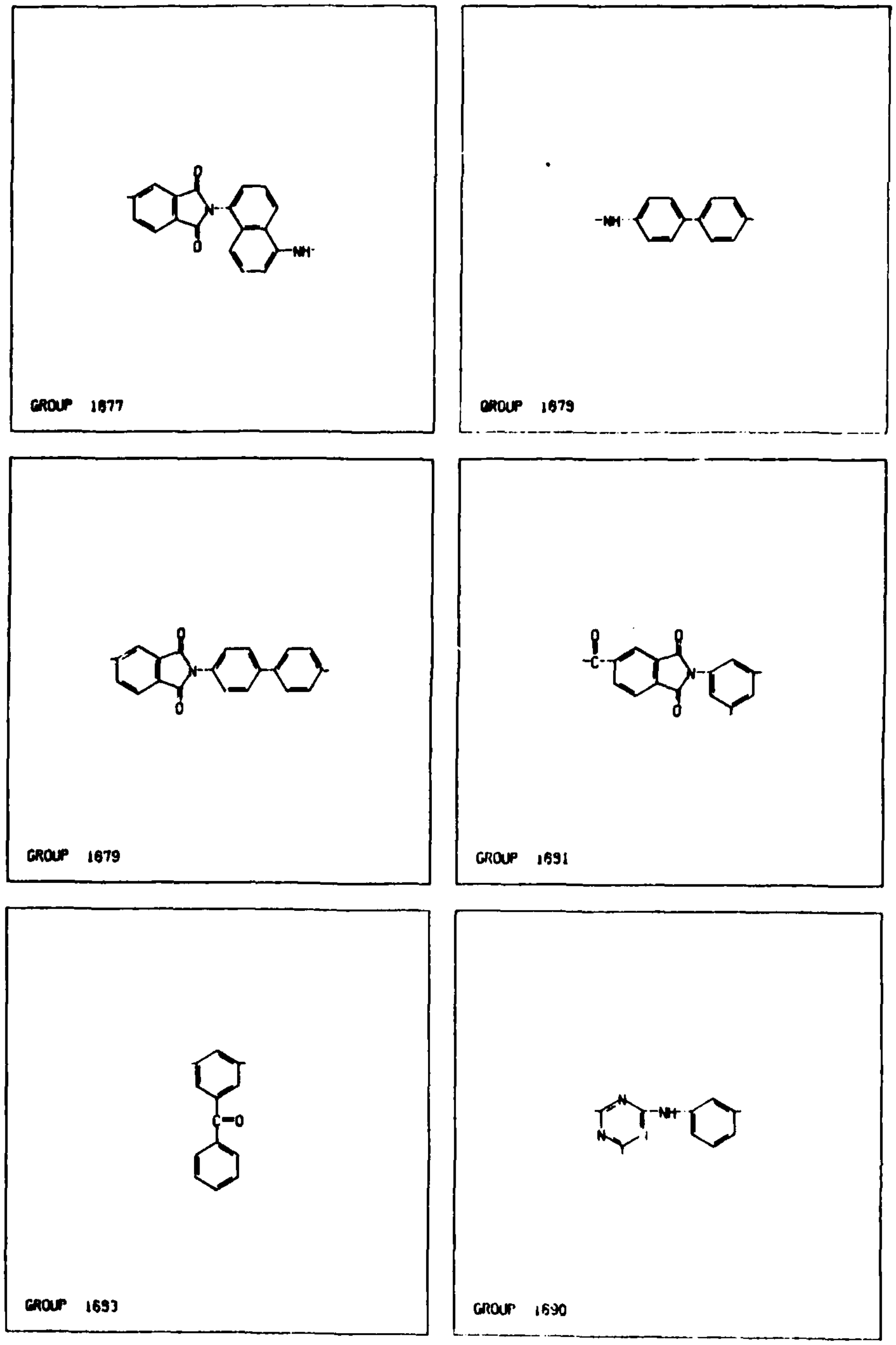


Fig 9 Examples of group drawings

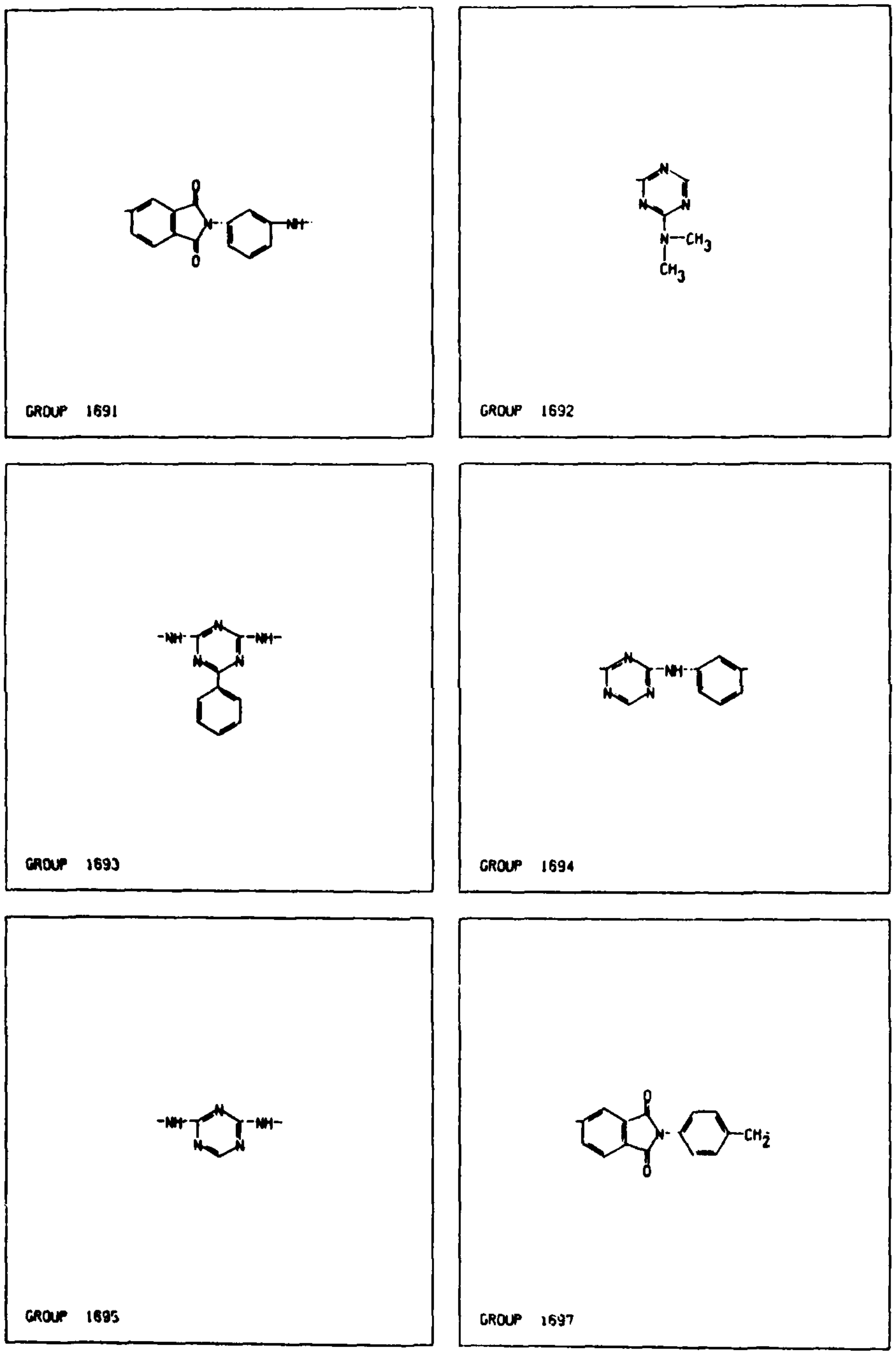


Fig 10 Examples of group drawings

Fig 11

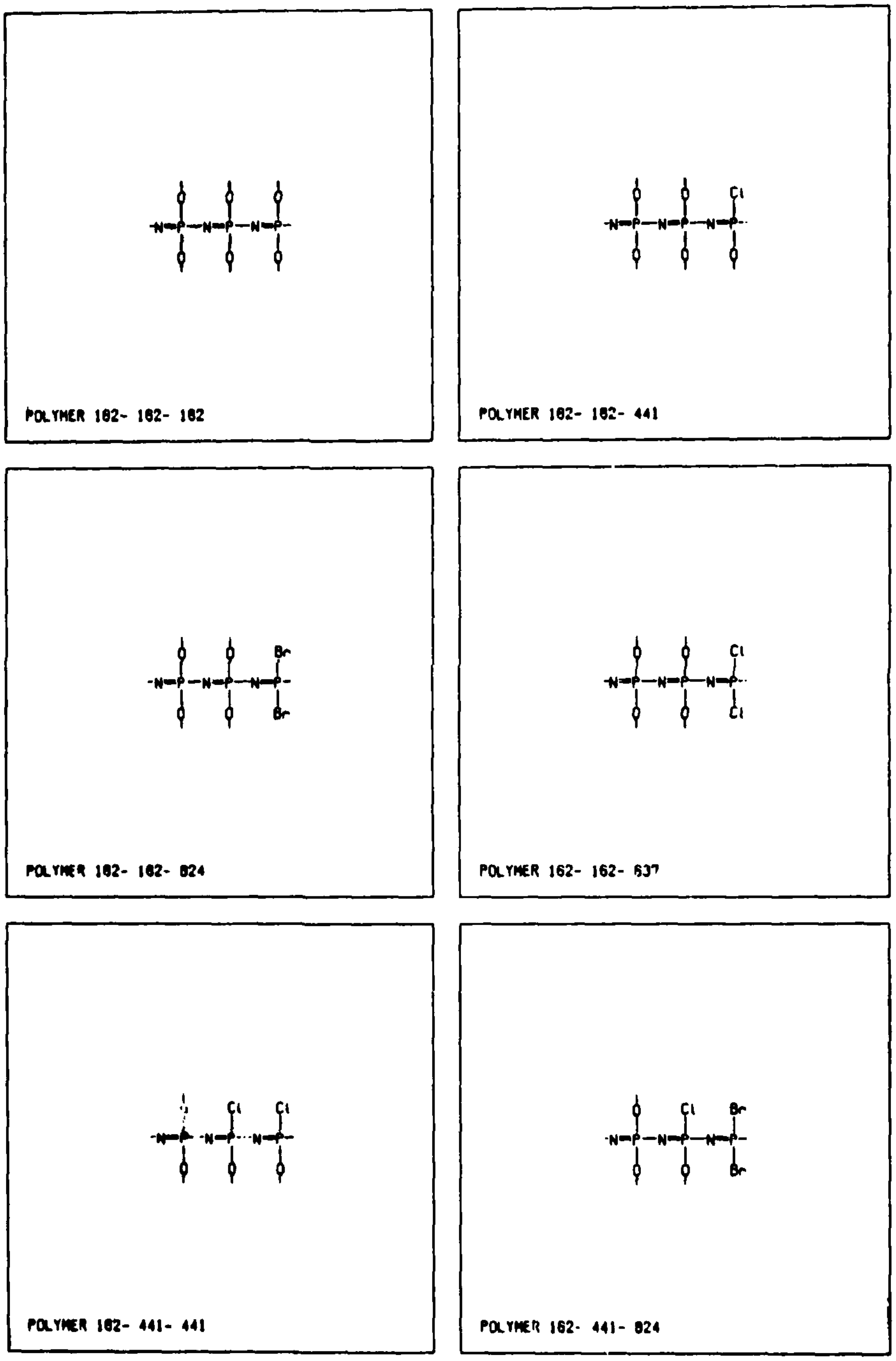


Fig 11 Examples of polymer drawings

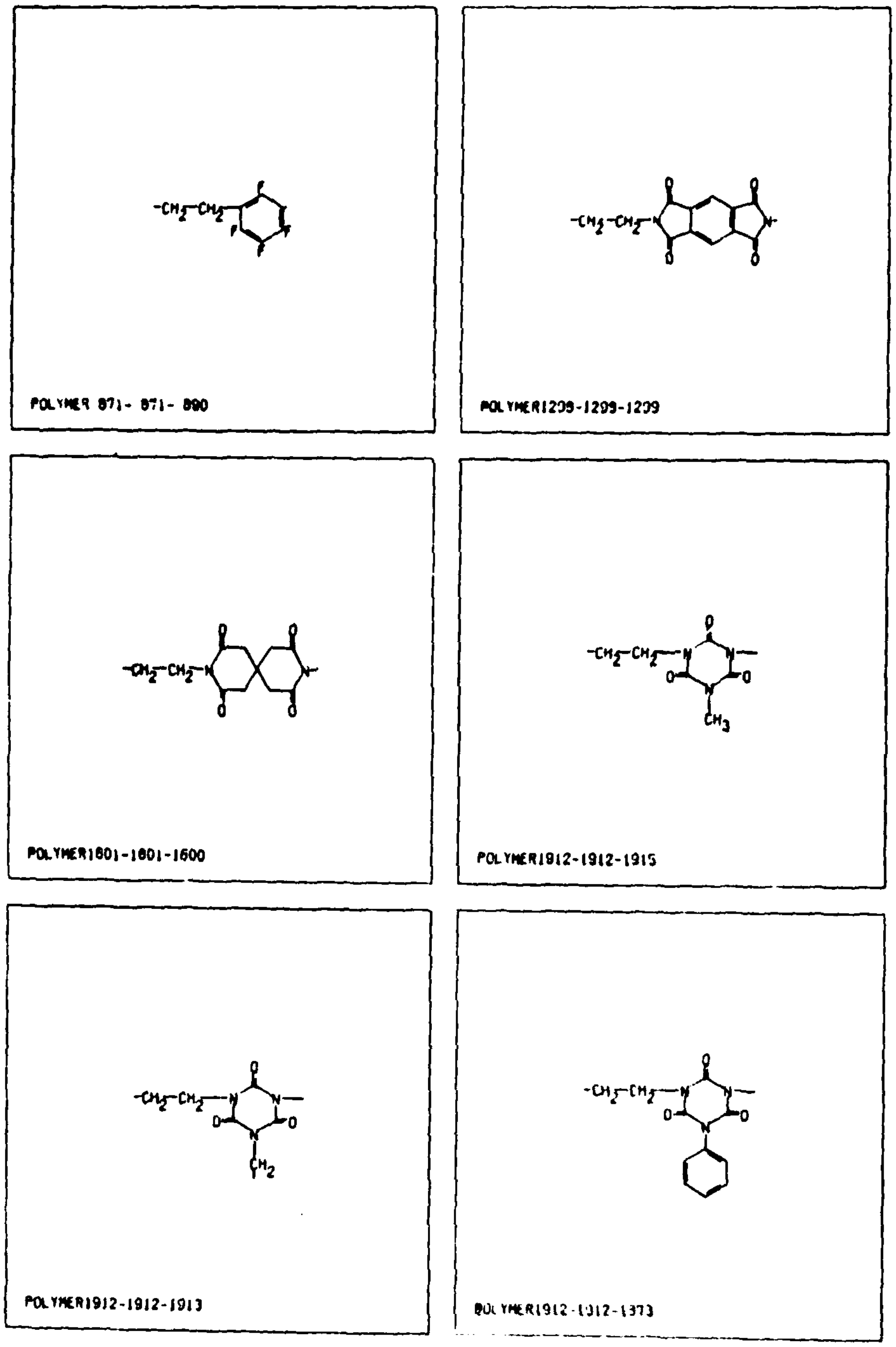
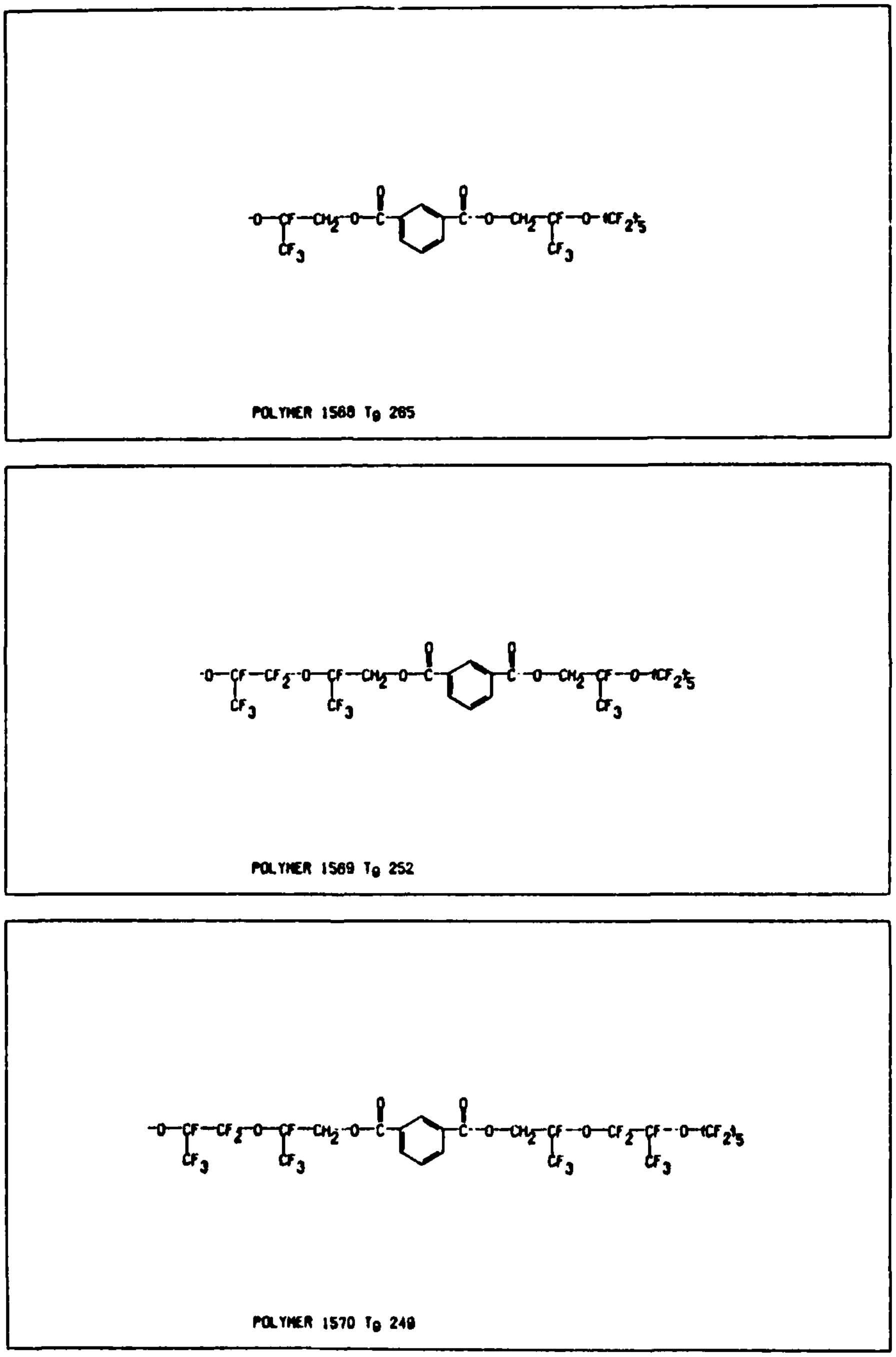


Fig 12 Examples of polymer drawings



POLYMER 1568 T_g 285

POLYMER 1569 T_g 252

POLYMER 1570 T_g 249

Fig 13 Examples of polymer drawings

REPORT DOCUMENTATION PAGE

Overall security classification of this page

UNLIMITED

As far as possible this page should contain only unclassified information. If it is necessary to enter classified information, the box above must be marked to indicate the classification, e.g. Restricted, Confidential or Secret.

1. DRIC Reference (to be added by DRIC)	2. Originator's Reference RAE TR 81030	3. Agency Reference N/A	4. Report Security Classification/Marking UNCLASSIFIED

5. DRIC Code for Originator 7673000W	6. Originator (Corporate Author) Name and Location Royal Aircraft Establishment, Farnborough, Hants, UK
5a. Sponsoring Agency's Code N/A	6a. Sponsoring Agency (Contract Authority) Name and Location N/A

7. Title
Production of chemical structure drawings using an interactive graphics system

7a. (For Translations) Title in Foreign Language

7b. (For Conference Papers) Title, Place and Date of Conference

8. Author 1. Surname, Initials Sharples, G.	9a. Author 2	9b. Authors 3, 4	10. Date February 1981	Pages 55	Refs. 3
11. Contract Number N/A	12. Period N/A	13. Project	14. Other Reference Nos. IT 171		

15. Distribution statement
(a) Controlled by – Head of Instrumentation and Trials Dept, RAE.
(b) Special limitations (if any) –

16. Descriptors (Keywords) (Descriptors marked * are selected from TEST)

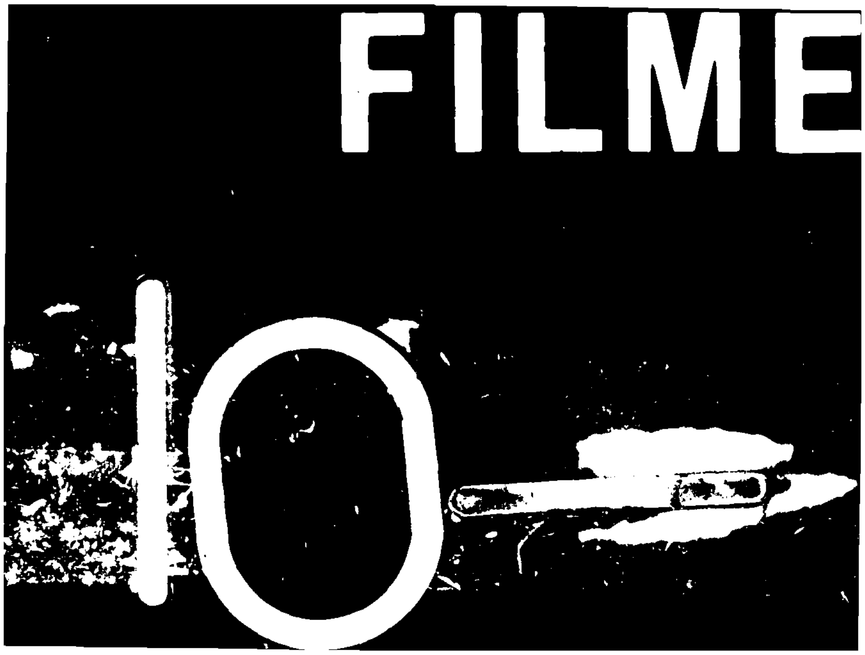
Graphic methods*. Computer programs*. Polymers*.

17. Abstract

This Report describes a series of computer programs which allows drawings of complex chemical structures to be displayed on a graphics terminal, to be modified using the graphical input devices and finally, to be drawn by an incremental plotter. The system thus provides a means for the interactive development of chemical structure diagrams and for the production of high quality drawings suitable for inclusion in published reports.

The system is based on the graphical definition of several hundred chemical groups. The structure of more complex compounds can be built up from these basic units and displayed to the user. Optional features of the system include variation of the scale of drawing, interactive modification of the drawings using a light pen and automatic detection and prevention of drawing overlaps.

F5910/1


FILME
10